Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 54 (2018)

Interactive Coding with Constant Round and Communication Blowup

Klim Efremenko* Elad Haramaty† Yael Tauman Kalai‡

Abstract

The problem of constructing error-resilient interactive protocols was introduced in the seminal works of Schulman (FOCS 1992, STOC 1993). These works show how to convert any two-party interactive protocol into one that is resilient to constant-fraction of error, while blowing up the communication by only a constant factor. Since these seminal works, there have been many followup works which improve the error rate, the communication rate, and the computational efficiency.

All these works assume that in each round each party sends a single bit, an assumption that may cause a substantial increase in the *round complexity*. Moreover, they assume that the communication complexity of the underlying protocol is *fixed* and a priori known.

In this work, we show how to convert any protocol Π, with *no a priori* known *communication bound*, into an error-resilient protocol Π', with comparable computational efficiency, that is resilient to constant fraction of adversarial error, while blowing up both the communication complexity and the *round complexity* by at most a constant factor. We consider the model where in each round each party may send a message of *arbitrary length*, where the length of the messages and the length of the protocol may be *adaptive*, and may depend on the private inputs of the parties and on previous communication. We consider the adversarial error model, where ϵ-fraction of the communication may be corrupted, where we allow each corruption to be an *insertion* or *deletion* (in addition to toggle).

In addition, we try to minimize the blowup parameters: In particular, we construct such Π' with $(1+\tilde{O}\left(\epsilon^{1/4}\right))$ blowup in communication and $O(1)$ blowup in rounds. We also show how to reduce the blowup in rounds at the expense of increasing the blowup in communication, and construct Π' where both the blowup in rounds and communication, approaches one (i.e., no blowup) as ϵ approaches zero. We give "evidence" that our parameters are "close to" optimal.

*Tel-Aviv University. klimefrem@gmail.com

†Amazon Research. seladh@gmail.com The research was conducted while the author was affiliated with Harvard University, Northeastern University and Microsoft Research.

‡Microsoft Research. yael@microsoft.com

ISSN 1433-8092

Contents

1 Introduction

Communication over a noisy channel is a fundamental problem in computer science, engineering and related fields. Starting from the seminal work of Shannon [Sha48], this problem of error-resilient communication has been extensively studied. Today, we have "good" error-correcting codes – ones that achieve constant information rate as well as constant error rate. The two main error models that were considered are the *stochastic* error model, where the errors are distributed according to some distribution (such as the binary symmetric channel), and the *adversarial* error model, where errors may occur adaptively and adversarially, so long as the prescribed error rate is not exceeded. This work considers the latter (stronger) adversarial error model. In addition, we consider (adversarial) insertion and deletion errors.

In a sequence of innovative works, Schulman [Sch92, Sch93, Sch96] initiated the study of error-resilience in the context of *interactive protocols*. Specifically, he considered the setting where two parties are interacting via a protocol over a noisy channel, where the noise could be stochastic or adversarial. Since Schulman's seminal works, there have been many followup works, that improve the error rate [BR11, GH13, GHS14, BE14, AGS16, EGH16], the information rate [KR13, Hae14, GH17], the computational efficiency [GMS11, BK12, BN13, BKN14], and very recently that are been beautiful works that generalize the error model of the adversary to allow insertions and deletions [BGMO16, HSV17, SW17]. There have also been several works that consider the multi-party setting [RS94, JKL15, BEGH16, GK17]. We refer the reader to [Gel17] for a fantastic survey on previous work on interactive coding. The focus of this work is on the 2-party setting and the adversarial error model.

All previous works assume that in the underlying protocol, in each round a *single* bit is sent. Thus, there is an inherent assumption that the round complexity is proportional to the communication complexity. In contrast, in cryptography and in distributed computing, protocols that consist of long messages are considered, and it is desirable to keep the round complexity as low as possible. In fact, much research (in both cryptography and distributed computing) focuses on reducing the round complexity of various protocols, as often the round complexity is the bottleneck, and not the communication complexity. We argue that since we consider *interactive* protocols, we should aim for error resilient protocols, that not only blow up the communication by at most a constant factor, but also blow up the number of rounds by at most a constant factor.

All previous works consider the "bit-by-bit" model, where each party sends a single bit in each round, and thus the protocols inherently have large round complexity. Therefore, we diverge from this "bit-by-bit" model, and consider the model where parties can send *arbitrarily long* messages in each round. Our model is the typical synchronous model used in cryptography and distributed algorithms. We elaborate on this model in Section 1.1.

Moreover, we emphasize that we do not assume that the communication (or round) complexity is fixed or a priori known. This is in contrast to all previous works, which assume that the communication (and round) complexity T is fixed and known in advance, and that the adversary can corrupt at most ϵT bits.[1] We note that such an assumption is often unrealistic and results in protocols where the communication complexity is always *worse-case*.

In this work, we allow the communication and round complexity to differ from execution to execution, depending on the inputs, or "types" of the parties, and construct an error-resilient protocol that preserves this per-execution communication (and round) complexity. We note

[1] We mention the work of Agrawal *et. al.* [AGS16], which does assume that the communication complexity of the underlying errorless protocol is a priori known. However, with the goal of maximizing the error rate, the communication complexity in the error-resilient protocol is not fixed and is not a priori known. We emphasize that in our work, we do not even assume that the parties a priori know the communication complexity in the underlying errorless protocol.

that the fact that we allow such adaptive (and not a priori known) communication length adds substantial technical difficulties to our work, which we elaborate on in Section 1.3.[2]

Our Results in a Nutshell. We show how to convert any protocol, where messages can be of arbitrary length, and where the communication and round complexity are not a priori known, into an error-resilient one, with comparable (computational) efficiency guarantees, that is secure against constant fraction of adversarial error, while incurring a constant blowup both to the communication complexity and to the round complexity. We allow the adversary not only to toggle with the bits of communication, but also allow the adversary to *insert* and *delete* bits. We elaborate on our communication model and error model in Section 1.1.

Moreover, we try to minimize the (constant) overhead in communication and rounds: In particular, we obtain $(1 + \tilde{O}(\epsilon^{1/4}))$ blowup in communication and $O(1)$ blowup in rounds. We also show how to reduce the blowup in rounds at the expense of slightly increasing the blowup in communication, and construct an error-resilient protocol where both the blowup in rounds and communication approaches one as ϵ approaches zero. We elaborate on our results (and on the exact parameters we obtain) in Section 1.2, we give a high-level overview of our techniques in Section 1.3, and give "evidence" that our parameters are "close to" optimal in Section 2.3 (after formally stating our main theorem in Section 2.2).

Our Technical Hurdles. The reader may at first think that dealing with short messages is the "hard case", since for long messages we can use standard error-correcting codes. We argue that this intuition is misleading. First, when considering adversarial error, applying an error-correcting code to each message separately does not help, since the entire message can be corrupted (even if the message is long), and indeed in this work we focus on adversarial error. We mention, however, that even for the case of stochastic error, dealing with messages of varying lengths, where some messages may be short while other messages may be long, is challenging.

Before explaining the difficulties that arise in this setting, we note that if we knew a priori the number of rounds and the communication complexity of the underlying protocol, then we could have "smoothed" it out perfectly, so that all the messages would have been of equal length,[3] and then we could have used a protocol (and analysis) from prior works.

Since we do not have such a bound, we cannot perfectly smooth out the underlying protocol. Nevertheless, we must somehow smooth out the protocol, since a party cannot send a long message before she is "sufficiently confident" that the transcript so far is correct, as otherwise, this long message will be wasted (even if the adversary does not corrupt it at all). Therefore, we "approximately" smoothen out the underlying protocol, by guaranteeing that each message is of length at least half and at most twice the length of the previous message. We refer the reader to Section 1.3 and Section 3 for details.

We mention that in order to minimize the blowup, we consider two small constants $\alpha, \beta > 0$ (that depend on the error rate) and guarantee that the length of each message is at least $\alpha\ell$ and at most $\frac{\ell}{\beta}$, where ℓ is the length of the message preceding it. This is not important for the high-level overview.

[2]We believe (though we haven't checked) that the tree-code based interactive coding schemes may easily be adapted to the setting where the communication complexity is not a priori bounded, by having each party construct an (infinitely growing) tree code. However, in tree-code based schemes the parties are computationally inefficient and there is a large blowup to the round complexity.

[3]This approach blows up the communication and round complexity by a constant factor. If the goal is to optimize this blowup (as we do in this work) then one cannot afford to perfectly smoothen out the protocol, in which case our techniques are needed.

We emphasize, that even after smoothing the underlying protocol, the length of the messages can still grow (or shrink) at an exponential rate, which brings rise to several challenges. For example, similar to many previous works (such as[Sch92, BK12, Hae14]), when a party realizes that there was an error she backtracks. In our setting we need to be extremely cautious when we backtrack. Note that the adversary can cause us to backtrack even though we are synchronized, by making us believe that we are out of sync. Previous works ensure that the adversary needs to invest enough error for such backtracking, and hence such "false" backtracking is costly for the adversary. However, in the case where messages are of varying length, this analysis becomes extremely delicate, since the adversary can corrupt a short message (by investing a small amount of his error budget), and thus falsely cause the parties to backtrack and delete a previous long message. Indeed, as opposed to previous protocols, we do not erase when we backtrack. Rather, we keep this transcript as "questionable". We refer the reader to Section 1.3 and Section 4 for details.

Moreover, when messages are of varying lengths, even if the protocol is (approximately) smooth, and even if we backtrack carefully, ensuring that the round complexity does not blow up, does not only require a careful (and significantly more complex) analysis, but also requires additional new ideas.

For example, the protocols in previous works, perform an equality test after every chunk of length d (for some parameter d), where in this equality test the parties check whether they are in sync by sending each other a hash of their transcript so far. In our setting, messages may be very long, and we cannot chop a message to chunks of d bits each, since this will blow up the round complexity. Instead, it is tempting to simply append to each message a hash of length that is proportional to the message length (e.g., append a hash of length $\lfloor \frac{\ell}{d} \rfloor$ to a message of length ℓ). However, as we show in Section 1.3 and Section 5, in order to ensure a constant blowup in round complexity, we must not only allow the length of the hash value to depend on the length of the message it is being appended to, but rather it should also depend on the length of the *entire history*. This is the case since if the protocol has messages of varying lengths, the adversary can corrupt a single long message, in a way that causes many hash collisions in future short messages. Thus, by corrupting one (long) message many rounds can be wasted.

In order to get around this problem, we allow the length of the hash to depend on the length of the entire history. Moreover, we consider randomized (i.e., seeded) hash function, where the party sends the hash value together with the hash seed, so that the adversary does not know which hash function will be used ahead of time. However, with a seed of length w one can hash messages of length at most 2^w, and the history may be longer than 2^w. Thus, in our scheme some of the seed is chosen ahead of time and some of the seed is chosen with each message. We refer the reader to Section 1.3 and Section 5 for details.

Moreover, the fact that the communication complexity is not a priori known creates an additional problem. Following previous works (such as [BK12, BKN14, Hae14]), we first construct a protocol in the *common random string* (CRS) model (this is done in Section 5), and then we remove the CRS (in Section 6). Removing the CRS in previous work was straightforward: First show that the CRS can be made relatively short (of size proportional to the communication complexity) by using a δ-biased source, and then argue that one of the parties can simply send the CRS using a (standard) error correcting code. In our case this cannot be done since we do not have an a priori bound on the communication complexity.

We give an overview on how we overcome the technical hurdles mentioned above in Section 1.3, but warn the reader that overcoming these challenges is quite difficult, and results in a very complex analysis.

We next explain our model in more detail.

1.1 Our Model

1.1.1 The Noiseless Model

We consider 2-party protocols, between two parties, Alice and Bob. In our model, at every round i, Alice and Bob do the following: Alice chooses $\ell_A(i) \in \mathbb{N}$ (greater than 0) and a message $m_A(i) \in \{0,1\}^{\ell_A(i)}$, based on her view of previous communication and her private input, and sends $m_A(i)$ to Bob. Similarly, Bob chooses $\ell_B(i) \in \mathbb{N}$ and a message $m_B(i) \in \{0,1\}^{\ell_B(i)}$, based on his view of previous communication and his private input, and sends $m_B(i)$ to Alice. At some round, one of the parties aborts, and both parties report an output.

More generally, we allow Bob's message in the i'th round to depend, not only on all previous communication and his private input, but also on Alice's message in the i'th round. This corresponds to the synchronous model where in each round i, Alice and Bob do not send their messages simultaneously, but rather first Alice sends her message and only then Bob sends his message (which may depend on Alice's message). This model is known as the message-passing model, and is the most common model used in cryptography (and distributed algorithms). We note that our results also apply to the synchronous simultaneous message model, and the choice of presenting our results in the synchronous message-passing model was due to the fact that we think that this model is more standard.

We emphasize that we allow the length of the messages in each round and the number of rounds to vary and to depend on previous communication. This models real world interactions where some conversations end fast, whereas others spark more interaction. We emphasize that all previous works considered the case where in each round a single bit is sent, but several of these works (such as [AGS13, GHS14]) considered adaptive protocols, where which party sends the bit may be a function of the transcript so far. Nevertheless, they all assumed that the (underlying errorless) protocol has an a priori fixed communication complexity.

We denote the input of Alice by x, and we denote the input of Bob by y. Note that a pair of inputs (x, y) define ℓ_A and ℓ_B for all rounds, and also define the number of rounds. Thus, in the noiseless setting, for any protocol we can define $\mathrm{CC}(x, y)$, which is the communication complexity of the protocol for the input pair (x, y). Similarly, we can define $\mathrm{R}(x, y)$, which is the number of rounds for the input pair (x, y).

1.1.2 The Noisy Model

In this work, we consider the adversarial error model, and assume that the adversary can corrupt any ϵ-fraction of the bits, for some a priori fixed small constant $\epsilon > 0$. We allow the adversary, not only to toggle with the bits, but he can also insert and delete bits.

In our model, where messages can be of arbitrary length, protecting protocols against insertions and deletions is extremely important, since otherwise the parties can securely encode information via the *length* of the messages. Specifically, in our model, where messages can be of varying lengths, one can trivially protect protocols against (adversarial) toggle corruptions while incurring only a constant factor blowup in the communication complexity, albeit an *unbounded blowup* in the round complexity, as follows: First convert the protocol to a protocol where each party sends a single bit in each round. Then, encode this bit as follows: If the bit is zero then encode it via a single bit (zero or one), and if the bit is 1 then encode it via any two bits. Upon receiving an encoded message the parties will decode without looking at the content of the message, but rather only by the length of the message.

We note that most previous work on interactive coding do not consider insertion and deletions. Indeed, in the synchronous model, where the parties send one bit per round, insertions and deletions are not interesting, since the parties "can tell" when an insertion or deletion

occurs.

An exception are the recent works of [BGMO16, HSV17, SW17]. These works consider insertion and deletions in the *asynchronous* model. More specifically, they consider an adversary who can insert a message from one party and delete a message from the other party, and thus cause the parties to be out-of-synch with regard to which round they are on (though similarly to previous work, they are in the bit-by-bit model, thus their protocols incur a large blowup to the round complexity, and they assume an a priori bound on the communication complexity).

In our work, we consider the *synchronous* model, where the parties always agree on the number of speaking alternations (which in our case is exactly the number of rounds). We emphasize, however, that the work of [HSV17] shows a generic method for converting any error-resilient protocol in the synchronized model into one that that is error resilient in the asynchronized model. We believe that one can use their approach (and in particular the use of a synchronization code [HS17]) to boost our result from the synchronous model to the asynchronous model.

In the adversarial error model, in our work and in all previous works, there is an a priori fixed constant ϵ and it is assumed that the adversary can corrupt at most ϵT bits, where T is the number of bits communicated. In most previous work, the value of T was assumed to be a priori known (and fixed). As mentioned above, in this work, we allow the length of the protocol to depend on previous communication. This models the real world setting, where we cannot a priori predict the length of our conversations, and it can depend on our private inputs (or on our "types").

In this model, care should be taken when defining the adversarial error model. One possibility is to allow the adversary to corrupt ϵT bits, where T is the number of bits that would have been transmitted assuming no error.[4]

We note, however, that with such a definition the adversary can use his ϵT bits of corruption budget, and cause the parties to abort prematurely. Namely, he can convince both parties that the other party is "boring" (i.e., that the other party has an input such that if they were executing the error-free protocol without error, the number of bits exchanged would have been less than ϵT). In such case both parties would abort prematurely and the adversary would "win".

Instead, we allow the adversary to corrupt only ϵ-fraction of the bits that were actually communicated. We note that a similar model was used in the work of Agrawal *et. al.* [AGS16], where their goal was to get optimal error-rate, and to that end, they considered error-resilient protocols with an adaptive speaking order and where the communication complexity may depend on the error pattern.[5] We emphasize that this adversarial model is stronger than alternative (natural) models, such as the the prefix model that allows the adversary to corrupt ϵ-fraction of any prefix of the transcript.

In this work, we also add a bound ϵ' on the number of rounds that the adversary can "fully" corrupt, where we say that a round is *fully corrupt* if the adversary corrupts more than δ-fraction of the bits, for some small constant δ (which depends on the error bound ϵ). We note that all previous works also had such a bound (implicitly), since in previous works there was no distinction between rounds and communication. In contrast, in our model, a bound on the number of bits corrupted does not imply a bound on the number of rounds that are fully corrupted. For example, consider the protocol in which there is one long message of length ℓ, followed by $\epsilon\ell$ messages, each consisting of a single bit. In such a protocol, not corrupting the

[4]We note that the adversary knows T since we assume (similarly to all previous work that consider the adversarial error model), that the adversary knows the private inputs of the parties.

[5]As mentioned above, the work of [AGS16] does assume an a priori known bound on the communication complexity of the underlying (errorless) protocol.

long message gives the adversary the budget to corrupt all the short messages.

We emphasize that bounding the number of rounds that are fully corrupted is necessary, since without such a bound, it is impossible to ensure a small blowup in round complexity. This argument is deferred to Section 2.3 where we give evidence to the optimality of our parameters.

1.2 Our Results

In what follows, we denote our error parameters by ϵ and ϵ', where ϵ corresponds to the fraction of corrupted bits, and ϵ' corresponds to the fraction of (fully) corrupted messages. We show that for any (small enough) constants $\epsilon, \epsilon' > 0$ there exist blowup parameters $\alpha, \alpha' > 0$ such that one can convert any protocol into an error resilient one (with respect to ϵ and ϵ'), with α blowup in communication, and essentially α' blowup in rounds (with an additional term that depends logarithmically on the communication complexity). We can set $\alpha' = O(1)$ we obtain a blowup of $\alpha = \tilde{O}\left(\epsilon^{1/4}\right)$ in communication complexity. Alternatively, one can set α, α' such that they both approach 0 as ϵ, ϵ' approach 0.

Our error-resilient protocol is randomized, even if the original protocol was deterministic. This is similar to all previous works that construct computationally efficient interactive coding schemes that are robust to adversarial error (starting with the work of [BK12]). Schulman [Sch93] (followed by many followup works) gave a deterministic interactive coding scheme, at the price of computational inefficiency. The parties in the error resilient scheme run in exponential time in T, where T is an upper bound on the length of the underlying protocol.[6] Recently, Gelles *et. al.* [GHK+16] gave a deterministic and efficient construction for the case of random error. However, constructing a deterministic interactive coding scheme that is resilient to adversarial error and is computationally efficient remains an interesting open problem.

We are now ready to state our main theorem. The most general theorem can be found in Section 2, and in what follows we present our theorem in a regime of parameters that we think is of particular interest.

In what follows, we let $t_{\min}$ denote the minimum value for which the underlying error-free protocol Π transmits at least $t_{\min}$ bits.

Theorem 1 (Main Theorem (informal))**.** *For any sufficiently small $\epsilon \geq 0$ and for any $\epsilon' \leq \epsilon^{1/4}$, there exist blowup parameters α and α', and a polynomial time probabilistic oracle machine S, such that the following holds. For any adversary $\mathcal{A}$ that corrupts at most ϵ-fraction of the bits of the simulated protocol $\Pi'_{\mathcal{A}}$ (which is the protocol Π' executed with the adversary $\mathcal{A}$), and "fully" corrupts at most ϵ'-fraction of the messages of $\Pi'_{\mathcal{A}}$, where $\mathcal{A}$ "fully" corrupt a message if he corrupts at least α^2-fraction of the bits of the message, we have the following guarantees.*

1. $\mathrm{CC}(\Pi'_{\mathcal{A}}) \geq t_{\min}$.
2. $\Pr\left[\mathrm{CC}(\Pi'_{\mathcal{A}}) > (1+\alpha)\mathrm{CC}(\Pi)\right] = \exp\left(-\mathrm{CC}(\Pi'_{\mathcal{A}})\right)$.
3. $\Pr\left[R(\Pi'_{\mathcal{A}}) > (1+\alpha')R(\Pi) + \alpha' \log \mathrm{CC}(\Pi)\right] = \exp\left(-R(\Pi'_{\mathcal{A}})\right)$.
4. $\Pr\left[(\mathsf{Output}(\Pi'_{\mathcal{A}}) \neq \mathsf{Trans}(\Pi))\right] = \exp\left(-\mathrm{CC}(\Pi'_{\mathcal{A}})\right)$.

Moreover, we can choose the parameter α, α' such that $\alpha = \tilde{O}(\epsilon^{1/4})$ and $\alpha' = O(1)$, or we can choose α, α' such that α and α' approach 0 as ϵ approaches 0.

[6] Braverman [Bra12] showed how to improve the parties' runtime to be sub-exponential in T.

The purpose of $t_{\min}$. In the theorem above, without adding the restriction that $CC(S^A, S^B) \geq t_{\min}$, the simulated protocol could have aborted as soon as more than ϵ-fraction of error was detected. In particular, if the first bit was noticeably corrupted, then the parties in the simulated protocol could have safely aborted. Thus, without adding the restriction that $CC(S^A, S^B) \geq t_{\min}$, this theorem does not even generalize previous works, which all assume an a priori fixed transcript size t and assume the adversary makes at most ϵt corruptions. Adding this restriction, gives the adversary an initial budget of $\epsilon t_{\min}$ corruption bits. Moreover, since the error probability is exponentially small in the actual transcript length, the requirement $CC(S^A, S^B) \geq t_{\min}$ guarantees a low error probability.

1.3 Overview of Our Techniques

In this section we give a high-level overview of the main ideas behind our construction and our analysis. In this overview, we do not focus on getting "optimal" parameters, and focus on constructing an error-resilient scheme that blows up the round and communication complexity by a constant factor. We note that all the conceptual ideas in this work are needed even to achieve constant overhead.

We start with an arbitrary protocol Π.

Smoothness. We first convert Π into a smooth protocol, with the property that after a message of length ℓ comes a message of length at most 2ℓ, and before a message of length ℓ comes a message of length at least $\ell/2$. We mention that in the actual protocol, to minimize the blowup in rounds and communication, we define (α, β)-smoothness, where α and β are functions of the error rate ϵ, and the guarantee is that after a message of length ℓ comes a message of length at least $\alpha\ell$ and at most $\frac{\ell}{\beta}$, and we show how to convert any protocol Π into an (α, β)-smooth protocol.

As mentioned above, the reason we need to smoothen Π is that otherwise, if after receiving a short message a party sends a long message, then the adversary by corrupting the short message, can cause the long message to be wasted, thus effectively allowing him to corrupt the long message by only using the budget needed to corrupt the short message.

Intuitively, we smoothen Π by instructing a party who wishes to send a long message after receiving a short message, to do so "cautiously", by sending the long message over several rounds, each time increasing the message length by at most a factor of 2.

To ensure that this does not cause a blowup to the round complexity, we make sure that a party does not send a short message after receiving a long one. Otherwise, suppose Alice always sends long messages (each of length ℓ) and Bob always sends single bit messages. Then by having Alice send her messages "cautiously", as explained above, the round complexity will blowup by a factor of $\log \ell$, which is too large. Instead, we instruct Bob to send longer messages, of length $\alpha\ell$, so that the adversary will need to invest enough budget to corrupt Bob's message; in particular, enough to allow Alice to send her length ℓ message safely.

We refer the reader to Section 3 for the formal definition of smoothness, and to Lemma 6 for how to convert a protocol into a smooth one.

From now on we assume the protocol Π is smooth, and show how to convert it into an error resilient one.

Message adversary. We first note that we can focus our attention only on adversaries, that rather than corrupting individual bits, corrupt messages, where the price of corrupting a message m is the maximum between the length of m and the length of the corrupted version

of m. If the adversary chooses to corrupt a message m then he may corrupt it adversarially, and if the adversary chooses not to corrupt a message then he cannot make any changes to it.

The reason we can focus on such adversaries is that we can easily convert any protocol that is resilient to errors made by message adversaries into a protocol that is error resilient to any adversary by applying an error correcting code to each message, and hence if only a small fraction of a message is corrupted (smaller than the allowed error rate) then this corruption can be ignored, since it is immediately corrected by the error correcting code. We use the error correcting code of Guruswami and Li [GL16], that is resilient to insertion and deletions, and has a minimal blowup of $1 + \tilde{O}(\sqrt{\epsilon})$ to the message length.

Thus, from now on, throughout this section, we ignore the layer of error correcting code, and consider only message adversaries.

1.3.1 The Protocol in the Ideal Hash Model

We first show how to convert any protocol Π into a protocol that is error resilient in the *Ideal Hash Model*. As in previous works (starting with the original work of [Sch92]), our starting point is the idea of using hashing to check for consistency. Namely, in the protocol Alice and Bob check equality of their partial transcripts, by sending to each other hashes of their partial transcripts.

In the Ideal Hash Model, we assume the existence of an "ideal" hash function, that is known to all parties and does not need to be communicated, and in the analysis we assume that the number of hash collisions is bounded, yet adversarially chosen (where the cost for each hash collision is proportional to the length of the hash value). We later elaborate on how we remove this ideal model assumption, by implementing this ideal hash using a real hash function.

For the sake of simplicity, throughout this overview we think of the parties appending to each message they send a hash of their transcript so far. We mention however, that in the actual protocol, since we want to optimize the communication blowup, we append a hash only to "long enough" messages, i.e., messages of length at least d, for a carefully chosen parameter $d \in \mathbb{N}$. In particular, we do not append a hash to short messages, and instead add a hash in every round that divides d (to take care of the case where all the messages are short).

Each party, upon receiving a message, first checks the consistency of the corresponding hash with its current transcript. If an inconsistency occurs, the parties enter a *correction mode.*

Correction Mode. In correction mode, the parties realize that their transcripts are inconsistent, and they need to rewind their transcript to a point where they believe they are consistent, yet without backtracking too much. Note that once an error is detected, the parties cannot simply rewind their transcript one round at a time, since the adversary can cause them to completely get out of sync. Moreover, they cannot send each other the round number they are currently simulating, as was done in [BK12], since this will blowup the communication by too much. Instead, we adapt the idea of backtracking to a "meeting point", an idea that was originated in [Sch92] and used in [Hae14]. For the sake of completeness, we explain this idea below.

Once the parties realize they are not in sync, they enter a correction mode, and once in error mode, they send two hashes of their transcript: One hash of the entire transcript, and the other of the transcript up until the second largest round. If a consistency was found they go back to the point of consistency. Otherwise, they send two hashes of their transcript until the largest, and second largest, round which is a multiple of 2. Again, if a consistency was found they go back to the point of consistency. Otherwise, in the i'th try, they send two hashes of their transcript until the largest, and second largest, round which is a multiple of 2^{i-1}.

In order to avoid the situation where the adversary invests $O(1)$ corruptions, and causes a party to go back 2^i steps, and thus lose 2^i bits of a possibly good transcript, the parties go back 2^i steps only after receiving roughly 2^i confirmations. The confirmations cannot be in a single round, since then the adversary could corrupt a single round and cause the parties to go back (possibly) 2^i rounds. Thus, instead these confirmations should span roughly 2^i rounds, and each party keeps a counter of how many confirmations it has.

One important missing piece is that they can be out of sync with respect to which are the meeting points. Thus, we also append to the message a hash of E, which denotes the number of rounds the party is in the error mode, and this length determines where the meeting points should be (which is roughly the power of 2 closest to E).

In previous works, once the parties backtrack, they erase the (seemingly) inconsistent transcript and continue to simulate the actual protocol. One important point where our protocol differs from all previous work, is that in our protocol the parties cannot afford to erase their (seemingly) inconsistent transcripts. This is due to the fact that the messages in the (seemingly) inconsistent part may be very long. For example, consider the case where the last message added to the transcript is of length 1, the one prior is of length 2, the one prior is of length 4, then length 8, and so on. Suppose no errors occurred and everything is consistent. The adversary can corrupt the hash appended to the short (1 bit) message, making the parties believe that their transcripts are inconsistent. The parties will backtrack, but the adversary will continue to make them believe that they are inconsistent, so that they erase i messages. This means erasing 2^i bits of communication, which is way more than the parties can afford to erase. Therefore, in our protocol, rather than erasing the (seemingly) inconsistent transcript, we keep it as questionable, and enter what we call a *verification mode*.

Verification Mode. In the verification mode, the parties simply test whether their questionable messages are consistent. They do this round-by-round, by sending a hash of the messages corresponding to each round. If their hashes agree, they mark the round as valid, and continue to the next round. If they arrive to a round where their messages do not agree, they don't immediately erase all the questionable transcript. Rather, they erase it only after they are "sufficiently" confident that they are inconsistent. To this end, they send longer and longer hashes until the number of bits of hash are proportional to the (seemingly) inconsistent transcript, and if the inconsistency persists then the parties erase their questionable transcript, and continue to simulate the underlying protocol.

This protocol is formally presented in Section 4, and the formal analysis in the Ideal Hash Model can be found in Section 4.2 and in Appendix B.

1.3.2 Our Protocol in the Shared Randomness Model

We next show how to implement the ideal hash functions with a specific hash function. To this end, we construct a function family $\mathcal{H} = \{h_x\}$, where each hash function h_x is associated with a (possibly long) seed x.

We consider the *shared randomness model*, where the parties are allowed to share a (possibly long) random string. We later show how to eliminate the need for shared randomness. But for now, we assume that the shared randomness is as long as we need. In particular, we use a different hash function (i.e, a different seed) for each equality test, and assume that the shared random string contains all these seeds. Since the length of the protocol is adaptive and not a priori bounded, the length of the common random string is also not a priori bounded.[7]

[7]We later show how we convert any such protocol in the *unbounded* shared randomness model into one that

We emphasize that the shared randomness (and in particular the seeds) are known to the adversary. Therefore the adversary, given a seed x, can try to skew the protocol and cause the parties to send many messages whose hashes collide.

Note that the adversary has $\binom{t}{\epsilon t} = 2^{\tilde{O}(\epsilon)t}$ different ways to corrupt the t bits of the communication. Thus, he can cause hash collisions in approximately $\tilde{O}(\epsilon)t$ bits. If we append each message of size ℓ with $O(\ell)$ bits of hash, the adversary will be able to cause hash collisions in messages with total volume of $\tilde{O}(\epsilon)t$, which is within the allowed error range. Indeed, our main challenge is to bound the number of *rounds* with hash collisions, a challenge that previous works did not need to deal with since in their setting, communication complexity and round complexity are equivalent.

If we a priori knew the length of the transcript t and the number of rounds R, then we could add $U = \frac{t}{R}$ bits of hash to each message, and since the adversary can cause only $\tilde{O}(\epsilon)t$ bits of hash collisions, the number of rounds in which the adversary can cause a hash collision, is bounded by $\tilde{O}(\epsilon)R$, which is again within our allowed error range.

Since we don't have such a bound, it is tempting to append to each message sent in round r a hash of length $U_r = \frac{t_r}{r}$, where t_r is the communication up to the round r. But the following example shows that such a padding does not suffice, and the adversary can still force too many rounds with hash collisions.

Consider a protocol that consists of $\tilde{O}(1/\epsilon)$ chunks such that chunk 0 consists of k single bit messages, and each chunk $i \neq 0$ consists of a single (long) message of length $2^i k$, followed by $\tilde{O}(\epsilon)k$ single bit messages. Note that in this case, the total number of hash bits in chunk i is $\approx 2^i$, and thus an adversary that corrupts the long message of this chunk can cause hash collisions in all the rounds of the chunk, resulting with a total of $O(R)$ rounds with hash collisions.[8]

To overcome this issue, the idea is to partition the protocol to chunks (which we call regimes), and append to each message a hash of length that is proportional to the average length of a message in the chunk. To be precise, we append to each message a hash of length $U_r = \max_{r'<r} \frac{t_r - t_{r'}}{r - r'}$. Unfortunately, in this case the total amount of hash bits being added can be as large as $t \log t$, which we cannot afford.

To overcome this issue, in each round, instead of sending all the U_r bits of hash, we send only a hash of these bits, where the seed of this (outer) hash is chosen using private randomness. Specifically, rather than sending $H_x(T)$ (which consists of U_r bits), the party chooses a random seed S, and sends $H_S(H_x(T))$, together with S. This reduces the number of bits being communicated from U_r to $\log U_r$. Since the adversary does not a priori know the private randomness chosen by the parties, he cannot corrupt the history to cause a hash collision in H_S in too many rounds. Moreover, since we saw that he cannot cause hash collisions in $H_x(T)$ in too many rounds, these hash functions are "safe". We note that a similar idea of using a randomized hash function was used by Haeupler [Hae14], for the sake of improving the rate of his interactive coding scheme. We refer the reader to Section 5 formal description of the hash function and to Appendix C.1 for the analysis of the number of hash collisions.

Finally, to conclude the analysis, we need to show that adding these (randomized) hash functions does not blow up the communication by too much. More precisely, one needs to show that $\sum_r \log U_r = O(t)$. This analysis is extremely delicate and requires several new ideas. This is done in Appendix C.2.

---

uses only private randomness.

[8] to be more precise, we need a long enough message at the end of the protocol to give the adversary enough budget to corrupt all of the long messages.

1.3.3 The Protocol in the Private Randomness Model

Finally, we show how to remove the need for shared randomness, while using only the private randomness of the parties. Namely, we show how to convert any protocol Π in the shared random string model, to one that uses only private randomness.

The basic idea is to follow the approach used in previous works (such as [BK12, Hae14]), and replace the long shared randomness with $2^{-O(T)}$-biased randomness, where T is an upper bound on the communication complexity. Such $2^{-O(T)}$-biased randomness can be generated using only $O(T)$ random bits. So, the basic idea is to send these $O(T)$ bits of randomness in advance, using an error correcting code. If we indeed had a bound T on the communication complexity, then this idea would work, and we would be done.

However, in our setting, we do not have an a priori bound on the communication complexity. In particular, if the communication complexity exceeds $O(T)$, then the adversary has the budget to corrupt more than $O(T)$ bits, and hence can completely corrupt the randomness s. We overcome this problem by sending more (and "safer") randomness as the communication complexity increases.

The protocol starts when one of the paries, say Alice, samples the shared random string $s_1 \in \{0,1\}^{O(t_{\min})}$ on her own (using her private randomness), and sends it to Bob. Then the parties execute Π with s_1 as the shared randomness. Once the communication complexity exceeds $O(t_{\min}/\epsilon)$, where ϵ is the corruption rate of the adversary, the random string s_1 is no longer "safe", and the the parties exchange a new random string s_2 of length $O(t_1)$, where t_1 is the current communication complexity. In addition to sending the new random string s_2 the parties also resend the previous random string s_1. The reason for resending previous seeds is that by resending the seeds the goal is to ensure that if one of the seeds was ever corrupted then the parties will "catch" the adversary, since the adversary does not have enough budget to continue to corrupt that seed, and the first time that he does not corrupt it, the parties will notice the inconsistency and abort, with the guarantee that the adversary performed too many errors.

In a similar way, after the communication complexity exceed $t_2 = O(t_1/\epsilon)$ a party will choose at random s_3 such that $|s_1| + |s_2| + |s_3| = t_2$, and will send (s_1, s_2, s_3), where t_2 is the current communication complexity, etc. As mentioned above, we ensure that if at any point, a message encoding randomness was decoded incorrectly, then eventually the paries will abort, and "catch" the adversary with injecting too many errors. This guarantee simplifies the analysis: Either at some point a randomness message was decoded incorrectly, in which case the adversary is "caught" with injecting too many errors, or all the parties always agree on the randomness, in which case correctness follows from the correctness of the underlying protocol in the shared randomness model.

A minor problem with the above idea is the following: a randomness message (s_1, s_2) may have been corrupted and converted into a protocol message, and a few rounds later a protocol message could have been corrupted and converted into the same randomness (s_1, s_2). To ensure that the parties will notice such corruption, we add to the randomness also the rounds $r_1, \ldots, r_k$ in which randomness were sent.

However, there is still a problem with the above idea, which is that in the early stage of the protocol, the shared random string has relatively large bias since it is generated using a short seed, and yet the adversary may have the budget to corrupt many bits, since the total communication may be large. It can be shown that such a powerful adversary can make too many hash collisions in the first part of the protocol.

To overcome this problem, we "enforce" that the adversary corrupts at most $O(\epsilon)$ fraction of any prefix of the protocol. To do so, in each time t_i, in addition to sending $(s_1, ..., s_{i+1})$

(together with $(r_1, \ldots, r_i)$), the parties send the transcript they have seen so far. If the parties detect that the adversary made significantly more than the allowed ϵ fraction of error, they abort, causing him to fail by exceeding his allotted corruption budget.

The formal protocol is described in Section 6, and its analysis is provided in Appendix D.

2 Our Results

In this section we present our main theorem, and give an intuitive argument for why our parameters seem to be optimal. We start by introducing notations and definitions that we use in our theorem, and throughout the manuscript.

2.1 Notations and Definitions

For any 2-party protocol $\Pi = (A, B)$, we denote by $\mathsf{Trans}(\Pi)$ the transcript of Π, which consists of all the messages exchanged throughout an execution of the protocol Π. We denote by $\mathsf{Output}(\Pi)$ the output of the parties after executing Π. We think of the protocol Π as being a deterministic protocol with no inputs. This is without loss of generality since we can always hard-wire the randomness and input into the protocol. We denote by $\mathrm{CC}(\Pi)$ the communication complexity of Π, and we denote by $R(\Pi)$ its communication complexity.

We consider simulators for simulating an interactive protocols. A simulator is a probabilistic oracle machine, that uses a protocol $\Pi = (A, B)$ as an oracle, and produces a new protocol $\Pi' = (S^A, S^B)$ that outputs the transcript of Π (even in the presence of error). For any adversary $\mathcal{A}$ we denote by $\Pi'_{\mathcal{A}}$ the protocol Π' executed with the adversary $\mathcal{A}$.

Definition 2. *We say that an adversary $\mathcal{A}$ corrupts at most ϵ-fraction of the bits of a protocol Π' if the number of corruptions made by $\mathcal{A}$ is at most $\epsilon \mathrm{CC}(\Pi'_{\mathcal{A}})$, where each corruption is either a toggle, an insertion or a deletion. The adversary $\mathcal{A}$ can be computationally unbounded, and its corruptions may depend arbitrarily on states of both parties in Π'.*

Definition 3. *We say that a message is γ-corrupted if the adversary corrupts at least γ-fraction of the bits of the message.*

We say that $f(x) = \tilde{O}(g(x))$ if there exists a $c \in \mathbb{N}$ such that $f(x) = O\left(g(x) \log^c\left(g(x)\right)\right)$ and we say that $f(x) = \tilde{\Omega}(g(x))$ if there exists $c \in \mathbb{N}$ such that $f(x) = \Omega\left(g(x) \log^{-c}\left(g(x)\right)\right)$.

2.2 Our Main Theorem

Theorem 4. *There exists a universal constant $\alpha_0 \geq 0$ such that for any blowup parameters $\alpha \leq \alpha_0$ and $\alpha' \leq 1$, there exist parameters $\epsilon = \tilde{\Omega}\left(\alpha^{3+\frac{1}{\alpha'}}\right)$, $\epsilon' = \tilde{\Omega}(\alpha\alpha'^3)$, and $\delta = \alpha^{O(1/\alpha')}$, and there exists a probabilistic oracle machine S, such that for any protocol $\Pi = (A, B)$, in which the parties always transmit at least $t_{\min}$ bits (even in the presence of error), and for any adversary $\mathcal{A}$ that corrupts at most ϵ-fraction of the bits of the simulated protocol $\Pi'_{\mathcal{A}}$, the protocol $\Pi'_{\mathcal{A}}$ (which is the protocol Π' executed with the adversary $\mathcal{A}$), satisfies the following properties.*

1. $\mathrm{CC}(\Pi'_{\mathcal{A}}) \geq t_{\min}$.
2. *There exists $t_0 = (1 + \tilde{O}(\alpha))\mathrm{CC}(A, B)$ such that for all $t > t_0$*

$$\Pr[\mathrm{CC}(\Pi'_{\mathcal{A}}) > t] \leq 2 \cdot 2^{-\delta t} ,$$

where the probability over the private randomness of S.

3. *There exists* $r_0 = (1 + O(\alpha'))\, R(A,B) + O\left(\frac{1}{\log \frac{2}{\alpha'}} \log \mathrm{CC}(A,B) + 1\right)$ *such that for any* $r \geq r_0$*, if at most* ϵ'*-fraction of the messages are* α^2*-corrupted, then*

$$\Pr\left[R(\Pi'_{\mathcal{A}}) > r\right] \leq 2 \cdot 2^{-\delta r} ,$$

where the probability over the private randomness of S*.*

4. *For any* $t > 0$*,*

$$\Pr\left[\left(\mathsf{Output}(\Pi'_{\mathcal{A}}) \neq \mathsf{Trans}(\Pi)\right) \ \wedge \ \left(\mathrm{CC}(\Pi'_{\mathcal{A}}) > t\right)\right] \leq 2 \cdot 2^{-\delta t} ,$$

where the probability over the private randomness of S*.*

5. S *is a probabilistic polynomial time oracle machine, and hence the computational efficiency of* S^A *and* S^B *is comparable to that of* A *and* B*, respectively.*

In Section 2.3 below, we give an intuitive argument for why our parameters seem to be optimal. Then, the rest of the manuscript is devoted to proving Theorem 4. Before, explaining our choice of parameters, in what follows, we give a high-level overview of the structure of the proof of Theorem 4.

Road Map. We first convert $\Pi = (A, B)$ into a smooth protocol Π_{smooth}. We show how this can be done in Section 3. Then, in Section 4, we show how to convert any smooth protocol Π_{smooth} into a protocol Π_{ideal}, which is error-resilient in the ideal hash model. In this model, we assume that the adversary is a "message adversary", which means that if he corrupts even a single bit of a message the price he pays for such a corruption is the length of the entire message (more precisely, the maximum between the length of the original message and the length of the corrupted version of it). Moreover, we assume that the number of hash collisions is bounded and adversarially chosen. We refer the reader to Section 4 for details.

In Section 5, we show how to convert Π_{ideal} into a protocol Π_{rand_1}, which is error resilient in the common random string model, assuming the adversary is a "message adversary". Loosely speaking, this is done by instantiating the ideal hash using public (and private) randomness. In Section 6, we show how to instantiate the common random string using private randomness, to obtain a protocol Π_{rand_2} that is error resilient against any "message adversary". Finally, we convert Π_{rand_2} into $\Pi' = (S^A, S^B)$, where Π' is the same as Π_{rand_2}, except that each message is sent encoded with the error correcting code that is resilient to insertions and deletions. In Section 7, we "put it all together" and prove that Π' is the error resilient protocol guaranteed in Theorem 4 above.

2.3 Intuition Behind our Parameters

In what follows, we give an intuitive argument for why our parameters seem to be optimal. We emphasize that this is by no means a proof of optimality, but rather an intuition for where these parameters came from.

As mentioned above, since messages can be of arbitrary length, and since we do not want to blow up the round complexity by much, we must use an error-correcting code that is resilient to (adversarial) insertions and deletions. To date, the maximal rate error-correcting code that is resilient to (adversarial) insertions and deletions is due to Guruswami and Li [GL16]. This code blows up the message length by $1 + \tilde{O}(\sqrt{\epsilon})$ and is resilient to ϵ fraction of errors.

Moreover, as argued in Section 1.3, in order to ensure a small blowup in communication our error-resilient protocol must be relatively "smooth". In other words, in the error-resilient

protocol, after a message of length ℓ we should not send a message much longer than ℓ, since then the adversary will corrupt the length ℓ message and as a result will cause the next long message to be obsolete. Suppose for simplicity (for now) that all the messages are all of the same length ℓ.

Suppose our protocol has blowup $1+\tilde{O}(\alpha)$ in communication complexity. Thus, we can use the error-correcting code of [GL16] that blows up the message length by at most $(1+\tilde{O}(\alpha))$. This code is resilient to α^2 fraction of errors. Thus, by corrupting $\alpha^2\ell$ bits of a message the adversary can make the next round completely obsolete. Since the adversary can corrupt ϵ-fraction of the bits, he can make $\frac{\epsilon}{\alpha^2}$-fraction of the rounds obsolete, which implies that it must be the case that $\frac{\epsilon}{\alpha^2} \leq \alpha$, which in turn implies that $\alpha > \epsilon^{1/3}$.

Note, however, that we cannot assume that all the messages are of the same length since this will blow up the round complexity by too much. And yet, as mentioned above, we do need to assume that the error-resilient protocol is somewhat "smooth", since otherwise the communication complexity will blow up by too much. Thus, we let $\beta > 0$ be a parameter, such that in the error-resilient protocol after a message of length ℓ comes a message of length at most $\beta^{-1}\ell$. Now, an adversary corrupting $\tilde{O}(\alpha^2 \cdot \ell)$ bits of a message can cause $\beta^{-1}\ell$ bits to be obsolete. Thus, intuitively, the parties may waste $\alpha^{-2} \cdot \beta^{-1}$ bits of communication per each corruption. This, together with the fact that the adversary has an ϵ-fraction of corruption budget and the fact that the communication blows up by at most $1+\tilde{O}(\alpha)$, implies that $\alpha^{-2} \cdot \beta^{-1} \cdot \epsilon < \alpha$, which in turn implies that

$$\alpha^3 \cdot \beta > \epsilon. \tag{1}$$

Therefore, on the one hand we would like to make β as large as possible, to improve the communication rate; on the other hand, increasing β blows up the round complexity. At first it seems that requiring this smoothing condition (i.e., that after a message of length ℓ comes a message of length at most $\beta^{-1}\ell$), will blow up the round complexity by too much. The reason is the following: Consider the real world example, where each message sent by Alice is of length ℓ, and each message sent by Bob is of length 1. Thus, to ensure that Alice is not wasting ℓ bits of communication due to a single error in Bob's message, we need to make the protocol smooth and have Alice send her message slowly, first sending the first β^{-1} bits, then after getting a bit of approval from Bob, Alice will send the next β^{-2} bits of her message, and so on. Thus, the number of rounds it will take Alice to send her message is roughly $\log_{\frac{1}{\beta}}(\ell)$. This will cause a blowup of roughly $\log_{\frac{1}{\beta}}(\ell)$ to the round complexity, which is way too much.

To avoid this blowup, we want to make sure that after a long message does not come a message which is too short, since short messages may cause a blowup to the round complexity (if the following message is long). However, this should be done while adding at most an α fraction to the communication complexity. Thus, we also smooth the protocol in the "other direction" and require that after a message of length ℓ comes a message of length at least $\alpha\ell$. Thus, going back to our example above, where Alice is talkative (sends messages of length ℓ) and where Bob sends messages of length 1, we first convert this to another protocol where Bob sends messages of length $\alpha\ell$. This does not change the round complexity at all, and changes the communication complexity by at most an α-factor. Now, we smoothen out this protocol, by having Alice, rather than sending her ℓ bit message in "one shot", she will first send $\beta^{-1}\alpha\ell$ bits, then send the next $\beta^{-2}\alpha\ell$ bits, and so on. Note that this will cause a blowup of $\log_{\frac{1}{\beta}}(\alpha^{-1})$ in the round complexity. Since we allow a blowup of at most α' to the round complexity (without taking into account the blowup due to error, or the additive term), we take β so that $\log_{\frac{1}{\beta}}(\alpha^{-1}) \leq \alpha'$, and thus we must take β such that $\beta \leq \alpha^{1/\alpha'}$. This together with Equation (1), implies that

$$\alpha^{3+1/\alpha'} > \epsilon,$$

as in Theorem 4 above.

It remains to explain the additive term in the round blowup and the multiplicative term that depends on the round error-rate ϵ'. For the latter, clearly, if ϵ'-fraction of the rounds were completely corrupted, these rounds need to be redone, and this incurs a blowup of $1+\epsilon'$ to the round complexity. As to the former, suppose the original protocol consists of a short message followed by a very long message, to make this protocol error resilient we will have to blow up the round complexity by essentially $\log_{\frac{1}{\beta}} \mathrm{CC}$, where CC is the communication complexity of the original protocol. This is the reason we have the log additive term in the round complexity.

Finally, we explain why $\epsilon' = \alpha \cdot \mathsf{poly}(\alpha')$. We note that for the purpose of our application (Theorem 1) the exact power of α' is not important. Consider a protocol that consists of a single bit per round. In this case we can effort to add a hash check only every α^{-1} rounds. In this case, the adversary can corrupt the first message of each chunk of α^{-1} rounds, which would render the entire chunk useless. Thus, a corruption of ϵ'-fraction of the rounds, may result with a round blowup of $\epsilon'\alpha^{-1} \leq \alpha'$, which implies that indeed $\epsilon' \leq \alpha\alpha'$.

3 Smooth Protocols

Throughout this section, we refer to "rounds" in a protocol as a one way communication. Namely, the number of rounds in a protocol is equal to the number of messages that are sent in the protocol. We note that in Section 4 we diverge from this interpretation, and refer to "rounds" as a back-and-forth communication between Alice and Bob. This inconsistency allows us to simplify the notation and the presentation. Note that these two interpretations can be interchanged, while incurring a blowup of at most 2 in the round complexity.

Let Π be an arbitrary 2-party protocol. We denote by m_r the messages sent in the r^{th} round in Π. In this section we show how to convert any protocol Π into a *smooth* protocol S^{Π}. In what follows we denote by M_r the message sent in the r^{th} round in S^{Π}.

Definition 5. *A protocol is (α,β)-smooth if for every round r the following holds:*

$$\alpha \cdot \max\{|M_{r-1}|,|M_{r-2}|,|M_{r-3}|\} \leq |M_r| \leq \frac{1}{\beta} \cdot \min\{|M_{r-1}|,|M_{r-2}|,|M_{r-3}|\} \tag{2}$$

Lemma 6. *For any $\alpha < \frac{1}{4}$ and $\beta \leq \frac{\alpha}{8}$, the following holds: Any protocol Π can be efficiently converted into an (α,β)-smooth protocol S^{Π} such that*

1. $\mathrm{CC}(S^{\Pi}) \leq \mathrm{CC}(\Pi) \cdot (1+50\alpha)$.
2. $R(S^{\Pi}) \leq R(\Pi) \cdot (1 + 8\log_{2\beta}\alpha) + 4\log_{\frac{1}{2\beta}} \cdot \mathrm{CC}(\Pi) + 4$
3. *If Π is computationally efficient then so is S^{Π}.*

We defer the proof of Lemma 6 to Appendix A.

Remark 7. *In Sections 4, 5, and 6, we show how to convert a smooth protocol into an error-resilient one. Similarly to previous error-resilient protocols in the literature, we will first pad the smooth protocol, and only then we convert the padded protocol into an error-resilient one. However, we will need to pad our protocol in a smooth way. This is done as follows: Suppose we want to pad our protocol with anywhere between L and $2L$ bits of 0's. Suppose that the last message in the smooth protocol is of length ℓ, then we add a message of length $\left\lfloor \frac{\ell}{\beta} \right\rfloor$, followed by a message of length $\left\lfloor \frac{\ell}{\beta^2} \right\rfloor$, and so on, until we add between βL and L bits, after which we add L bits (if we haven't added so already).*

Note that such a padding results in a smooth protocol, where the communication complexity increases by at least L and at most $2L$ bits. The number of additional rounds required to do this padding is at most $\log_{\frac{1}{\beta}} L + 1$.

From now on, when we say that we convert a protocol Π to a smooth protocol, we assume that the resulting smooth protocol is padded appropriately.

4 Interactive Coding in the Ideal Hash Model

In this section, we show how to convert any protocol Π into an error resilient protocol, and analyze its properties in the **Ideal Hash Model**. This model assumes the existence of an ideal hash. In our protocol, Alice and Bob check equality of their partial transcripts, by sending to each other hashes of their partial transcripts. In this section, we consider the Ideal Hash Model, where when we analyze the communication complexity of the protocol we do not take into account the length of the hash values, and simply assume that the number of hash collisions is bounded, yet adversarially chosen. (We explain how we bound the number of collisions below). In Sections 5 and 6, we show how to remove this ideal model assumption, by implementing this ideal hash using a real hash function. In these sections, we use hash values that are short enough so the communication blowup is small, and yet we prove that with high probability the amount of hash collisions is bounded.

Moreover, we consider an adversary that either leaves a message (and corresponding hash) intact, or "fully" corrupts it. More precisely, we say that the hash is corrupted if and only if a collision occurs. In the analysis of this ideal error-resilient protocol, we say that a message is corrupted if the adversary corrupts any bit of the message (or if he corrupts the corresponding hash). We let the budget of corrupting a message be the maximum between the original message length, and the corrupted one. In particular, even if the adversary corrupted a single bit of a long message of length n (or if he corrupts only the hash corresponding to this message), we count it as n corruptions. We recall that the reason for this budgeting is that in our actual error-resilient protocol we will apply the error correcting code of Guruswami and Li [GL16] to each message (and hash) separately. Thus, in order to corrupt a message, the adversary will need to corrupt a constant fraction of the bits in the message. We refer the reader to Section 7 for details.

In what follows, we set

$$\alpha \leq 0.01 \quad , \quad \alpha' \leq 1 \ , \ \ d \geq \frac{1}{\alpha} \ \text{ and } \ \beta \leq \min\left\{\alpha^{\frac{1}{\alpha'}}, \frac{1}{5\alpha d^2}\right\}. \tag{3}$$

We assume for simplicity that α^{-1} and β^{-1} are integers. We assume without loss of generality, that the underlying protocol Π is (α, β)-smooth. This is without loss of generality since by Lemma 6, we can convert Π to an (α, β)-smooth protocol while increasing its communication complexity by a multiplicative factor of $(1 + O(\alpha))$, and increase the number of rounds by a multiplicative factor of $(1 + O(\alpha'))$ and an additive factor of at most $\log \mathrm{CC}(\Pi)$, as desired.

4.1 The Protocol

We note that this (ideal) protocol is quite similar to the error-resilient protocol of Haeupler [Hae14]. The main difference being that we need to first convert the protocol into a smooth one (whereas the protocol considered in [Hae14] is perfectly smooth since in each round each party sends a single bit to the other party). Moreover, and more importantly, since our protocol is not perfectly smooth, when the parties backtrack, they do not erase the questionable transcript (since the messages in the questionable part may grow exponentially). Instead, the

parties keep this transcript as questionable, and enter a "verification" state where they check consistency round-by-round. We note that in [Hae14] the questionable transcript is simply erased.

In what follows, we present the (error-resilient) protocol only from Alice's perspective. Bob's perspective is symmetric. During the (error-resilient) protocol, Alice has a private variable T_A, which she believes to be a prefix of the transcript she is trying to reconstruct. T_A is initiated to $\emptyset$. We denote by m_A the message that Alice sends in the error resilient protocol.

In what follows, we define all the other notations (in addition to m_A and T_A) that are used in the protocol description:

$$S_A, R_A, \ell_A, \ell^+, \ell^-, w_A, R_A^{(1)}, R_A^{(2)},$$

where all of these variables are defined as functions of T_A and m_A.

From now on we think of each round as consisting of consecutive two messages: a message sent by Alice and a following message sent by Bob. We note that this diverges from the way we defined rounds in Section 3, where we thought of each round as containing a single message (sent by one party). The only reason for this inconsistency is that it is more convenient in terms of notations. It is important to note that this is only a notational convenience and does not affect our final result in any way.

For each variable used in our protocol

$$v_A \in \{T_A, m_A, S_A, R_A, \ell_A, \ell^+, \ell^-, w_A, R_A^{(1)}, R_A^{(2)}\},$$

we denote by $v_{A,r}$ the value of v_A that Alice uses when sending her round r message, and we occasionally omit r when it is clear from the context.

- During the protocol Alice has a state

  $$S_A \in \{\text{Simulation}, \text{Verification}\} \cup \mathbb{N}.$$

  Loosely speaking, Alice is in a Simulation state when she believes that the transcript T_A that she is holding is indeed a prefix of the correct transcript.

  If $S_A \in \mathbb{N}$ then we say that Alice is in a Correction state. If Alice is in Correction state, then S_A is the first round (in the error-resilient protocol) that Alice has entered this state. Alice enters a Correction state when she thinks her beliefs are wrong (for example, when the hashes indicate that T_A and T_B are inconsistent). During this state, Alice tries to go back to an earlier round in the transcript (corresponding to the original protocol) which she believes to be correct. We denote this round by R_A. Alice will continue the simulation from $T_A[R_A]$, which denotes the truncated transcript of T_A to round R_A. As mentioned above, as opposed to the protocol of Haeupler [Hae14], in our protocol, she does not delete the suffix of T_A, and rather she keeps this suffix as questionable. The reason she does not erase this questionable suffix, is that it may be the correct suffix (and the only reason it is questioned is due to an error), and in this case it may be too expensive to delete and reconstruct, since in our case the messages in the questionable suffix may grow at an exponential rate.

  After a Correction state, Alice either enters another Correction state or enters a Verification state, where she decides whether to completely delete, partially delete, or keep, the questionable suffix. After the Verification state (assuming there were no errors), Alice enters Simulation state again.

  We define $r - S_A$ to be zero, when $S_A \in \{\text{Simulation}, \text{Verification}\}$.

- As mentioned above, R_A denotes the round in T_A that Alice simulates. If $S_A = \text{Simulation}$ then R_A is equal to the number of rounds in T_A.
- Let $m_{A,r}$ be the message that Alice sends in round r (of the error resilient protocol), and let $\ell_{A,r}$ denote its size. Let $m_{B,r}$ be the message that Alice received from Bob in round r, and let $\ell_{B,r}$ denote its size. We define $\ell_{\max,r} = \max\{\ell_{A,r}, \ell_{B,r}\}$. Note that if Bob's message was corrupted then $\ell_{B,r}$ may be arbitrarily large. However, our (error-resilient) protocol has the property that if Bob's message was not corrupted then $\ell_{B,r} \leq \frac{\ell_{A,r}}{\beta}$.

  We define
  $$\ell_r^+ = \min\left\{\frac{\ell_{A,r}}{\beta}, 2\ell_{\max,r}\right\}$$
  and
  $$\ell_r^- = \min\left\{\frac{\ell_{A,r}}{\beta}, \max\left\{\beta^{-1}, \alpha\ell_{\max,r}\right\}\right\}.$$
- Let $w_{A,r}$ be $2^{\lfloor \log(r - S_A) \rfloor}$ if $S_{A,r} \in \mathbb{N}$, and let $w_{A,r}$ be 0 otherwise. In other words, $w_{A,r}$ is the number of rounds that the party has been in Correction state, rounded to the closest power of two.
- If $w_A = 0$ then let $R_A^{(1)} = R_A$. Otherwise, let $R_A^{(1)} < R_A$ be maximal that divided w_A.
- $R_A^{(2)} \triangleq R_A^{(1)} - w_A$.
- In the protocol, at each round r, Alice sends hashes to Bob if and only if $r = 0 \pmod d$ or $\ell_{A,r} \geq d$, in which case she sends five hashes, one hash corresponding to each of the following strings:
  $$\left(T_A[R_A], T_A[R_A + 1], T_A[R_A^{(1)}], T_A[R_A^{(2)}], S_A\right).$$
  We note that if R_A is equal to the number of rounds in T_A, then Alice will not know the partial transcript $T_A[R_A + 1]$. In this case we define $T_A[R_A + 1] = T_A[R]$.

Alice in round r. Upon receiving a message from Bob, parse the message as

$$(m_{B,r-1}, H(T_{B,r-1}[R_{B,r-1}]), H(T_{B,r-1}[R_{B,r-1} + 1]),$$
$$H(T_{B,r-1}[R_{B,r-1}^{(1)}]), H(T_{B,r-1}[R_{B,r-1}^{(2)}]), H(S_{B,r-1})).$$

We assume that in this ideal model, parsing is easy. When we implement this ideal hash function in Section 5, we will make sure that indeed Alice will be able to parse correctly (assuming the message was not corrupted). Denote the size of $m_{B,r-1}$ by $\ell_{B,r-1}$.

1. If $S_{A,r-1} = \text{Simulation}$ then do the following:
   (a) If a hash was sent by Bob (i.e., if $\ell_{B,r-1} \geq d$ or d divides $r - 1$) then check that
   $$H(S_{B,r-1}) = \text{Simulation} \quad \text{and} \quad H(T_{A,r-1}[R_{A,r-1}]) = H(T_{B,r-1}[R_{B,r-1}]).$$
   (b) Check that the (partial) transcript $(T_{A,r-1}[R_{A,r-1}], m_{A,r-1}, m_{B,r-1})$ satisfies the (α, β)-smoothness condition.
   (c) If one of these conditions does not hold then let
   - $m_{A,r} = 0^{\ell_{r-1}^-}$

- $S_{A,r} = r$
- $(T_{A,r}, R_{A,r}) = (T_{A,r-1}, R_{A,r-1})$.

(d) Else, set

- $T_{A,r} = (T_{A,r-1}, m_{A,r-1}, m_{B,r-1})$
- $R_{A,r} = R_{A,r-1} + 1$
- $m_{A,r} = \Pi(T_{A,r})$
- $S_{A,r} = S_{A,r-1} =$ Simulation.

2. If $S_{A,r-1} =$ Verification then check if all the following conditions hold:

(a) $|m_{B,r-1}| \geq \beta^{-1}$.
(b) $H(S_{A,r-1}) = H(S_{B,r-1})$
(c) $H(T_{A,r-1}[R_{A,r-1}]) = H(T_{B,r-1}[R_{B,r-1}])$.

If one of these conditions does not hold then let

- $m_{A,r} = 0^{\ell^-_{r-1}}$
- $S_{A,r} = r$
- $(T_{A,r}, R_{A,r}) = (T_{A,r-1}, R_{A,r-1})$.

Else, do the following:

(a) If number of rounds in $T_{A,r-1}$ is greater than $R_{A,r-1} + 1$, and

$$H(T_{A,r-1}[R_{A,r-1} + 1]) = H(T_{B,r-1}[R_{B,r-1} + 1]),$$

then let

- $m_{A,r} = 0^{\ell^-_{r-1}}$
- $R_{A,r} = R_{A,r-1} + 1$
- $(S_{A,r}, T_{A,r}) = (S_{A,r-1}, T_{A,r-1})$.

(b) Else, if $m_{A,r-1} = m_{B,r-1} = 1^\ell$ for some $\ell > |T_{A,r-1}| - |T_{A,r-1}[R_{A,r-1}]|$, then set

- $T_{A,r} = T_{A,r-1}[R_{A,r-1}]$
- $R_{A,r}$ be the number of rounds in $T_{A,r}$
- $m_{A,r} = \Pi(T_{A,r})$
- $S_{A,r} =$ Simulation.

(c) Else, if $\ell^+_{r-1} > |T_{A,r-1}| - |T_{A,r-1}[R_{A,r-1}]|$ then let

- $m_{A,r} = 1^{\ell^+_{r-1}}$
- $(T_{A,r}, R_{A,r}, S_{A,r}) = (T_{A,r-1}, R_{A,r-1}, S_{A,r-1})$.

(d) Else, let

- $m_A = 0^{\ell^+_{r-1}}$
- $(T_{A,r}, R_{A,r}, S_{A,r}) = (T_{A,r-1}, R_{A,r-1}, S_{A,r-1})$.

3. Else, do the following:

(a) Compute the values v_r^0, v_r^1, v_r^2 as follows:
If $H(S_{A,r-1}) \neq H(S_{B,r-1})$ then set $v_r^0 \leftarrow v_{r-1}^0 + 1$.
Else, if $H(T_{A,r-1}[R_{A,r-1}^{(1)}]) \in \left\{H(T_{B,r-1}[R_{B,r-1}^{(1)}]), H(T_{B,r-1}[R_{B,r-1}^{(2)}])\right\}$ then set $v_r^1 \leftarrow v_{r-1}^1 + 1$.
Else, if $H(T_{A,r-1}[R_{A,r-1}^{(2)}]) \in \left\{H(T_{B,r-1}[R_{B,r-1}^{(1)}]), H(T_{B,r-1}[R_{B,r-1}^{(2)}])\right\}$ then set $v_r^2 \leftarrow v_{r-1}^2 + 1$.

(b) If $r - S_{A,r-1}$ is not a power of 2,[9] then set
- $m_{A,r} = 0^{\ell_{r-1}^-}$
- $(T_{A,r}, R_{A,r}, S_{A,r}) = (T_{A,r-1}, R_{A,r-1}, S_{A,r-1})$.

(c) Else, if $v_{r-1}^0 > \frac{1}{2}(r - S_{A,r-1})$ then let
- $S_{A,r} = r$
- $m_{A,r} = 0^{\ell_{r-1}^-}$
- Set $v_r^0 = v_r^1 = v_r^2 = 0$.

(d) Else, if $v_r^1 > \frac{1}{4}(r - S_{A,r-1})$ then let
- $S_{A,r} =$ Verification
- $R_{A,r} = R_{A,r-1}^{(1)}$
- $m_{A,r} = 0^{\ell_{r-1}^-}$
- Set $v_r^0 = v_r^1 = v_r^2 = 0$.

(e) Else, if $v_r^2 > \frac{1}{4}(r - S_{A,r-1})$ then let
- $S_{A,r} =$ Verification
- $R_{A,r} = R_{A,r-1}^{(2)}$
- $m_{A,r} = 0^{\ell_{r-1}^-}$
- Set $v_r^0 = v_r^1 = v_r^2 = 0$.

(f) Else, set $v_r^1 = v_r^2 = 0$, and let
- $m_{A,r} = 0^{\ell_{r-1}^-}$
- $(v_r^0, R_{A,r}, S_{A,r}, T_{A,r}) = (v_{r-1}^0, R_{A,r-1}, S_{A,r-1}, T_{A,r-1})$.

Send $m_{A,r}$, and if $r = 0 (\text{mod } d)$ or $|m_{A,r}| \geq d$ then append to $m_{A,r}$ also

$$\left(H(T_{A,r}[R_{A,r}]), H(T_{A,r}[R_{A,r}+1]), H(T_{A,r}[R_{A,r}^{(1)}]), H(T_{A,r}[R_{A,r}^{(2)}]), H(S_{A,r})\right).$$

Remark. Bob behaves identically to Alice, except in Steps 1 and 2b, when Bob computes his next message corresponding to the underlying protocol Π, he computes it by $m_{B,r} = \Pi(T_{B,r}, m_{A,r})$, whereas recall that Alice computed it by $m_{A,r} = \Pi(T_{A,r})$.

4.2 Analysis

Terminology. In what follows, we introduce terminology that we use in the analysis.

We allow the adversary to create collisions in the ideal hash function, in which case we say that the hash was corrupted. We say that a message is corrupted if the adversary corrupts any bit of the message, or corrupts the associated hash. We define the budget of corrupting a

[9] Recall that we define $r - S_A = 0$ if $S_A \in \{\text{Simulation}, \text{Verification}\}$, and we consider 0 to be power of 2.

message m to be the maximum between the length of m and the length of the corrupted version of m. Thus, even if the adversary corrupts a few bits of a long message of length n (or corrupts the associated hash), then we count it as n corruptions. On the other hand, if the adversary corrupted a single bit message by converting it into a long n-bit message, then we count it as n corruptions.

We analyze the correctness of the (error-resilient) protocol assuming a bound on these message corruptions.

Definition 8. *We say that the corrupted messages have* volume e *if the sum of lengths of corrupted messages (where each such length is the maximum between the length of the original message and the length of the corrupted version of it) is e.*

Using this terminology we prove the following theorem.

Theorem 9. *Let $\Pi = (A, B)$ be any (α, β)-smooth protocol, and let $\Pi' = (S^A, S^B)$ be the simulated protocol defined above. Let $\mathcal{A}$ be any adversary in Π', who corrupts at most e' messages of total volume of at most e. Then, the protocol Π', executed with the adversary $\mathcal{A}$, denoted by $\Pi'_{\mathcal{A}}$, satisfies the following.*

1. $CC(\Pi'_{\mathcal{A}}) \geq t_{\min}$, *where $t_{\min}$ is a lower bound of the communication of any instance of Π.*
2. $CC(\Pi'_{\mathcal{A}}) \leq CC(A, B) + 18\beta^{-1}e + 20d\beta^{-1}e'$.
3. $R(\Pi'_{\mathcal{A}}) \leq R(A, B) + 906d \log \frac{1}{\beta} e'$.
4. *The parties outputs transcripts of size at most* $\mathrm{CC}(\Pi'_{\mathcal{A}})$ *that agree with Π on the first*

$$CC(\Pi'_{\mathcal{A}}) - 18\beta^{-1}e - 20d\beta^{-1}e' \ ,$$

   many bits.

5. *S is a polynomial time oracle machine.*

Remark 10. *We will apply Theorem 9 with an adversary $\mathcal{A}$ that corrupts at most $e' = O(\min\{\epsilon' \cdot R(\Pi'_{\mathcal{A}}), \frac{\epsilon}{d} CC(\Pi'_{\mathcal{A}})\})$ messages of total volume at most $e = O(\epsilon \cdot CC(\Pi'_{\mathcal{A}}))$, where $\epsilon \leq O(\alpha \cdot \beta)$ and $\epsilon' \leq \frac{\alpha'}{d \cdot \log \beta^{-1}}$. Thus,*

$$\begin{aligned}
&CC(\Pi'_{\mathcal{A}}) \leq \\
&CC(A, B) + 18\beta^{-1}e + 20d\beta^{-1}e' \leq \\
&CC(A, B) + 0(\beta^{-1}\epsilon \cdot CC(\Pi'_{\mathcal{A}})) + O\left(d\beta^{-1}\frac{\epsilon}{d} CC(\Pi'_{\mathcal{A}})\right) \leq \\
&CC(A, B)(1 + O(\alpha)),
\end{aligned}$$

and

$$\begin{aligned}
&R(\Pi'_{\mathcal{A}}) \leq \\
&R(A, B) + 906d \log \frac{1}{\beta} e' \leq \\
&R(A, B) + O(d \log \frac{1}{\beta} \epsilon' \cdot R(\Pi'_{\mathcal{A}})) \leq \\
&R(A, B)(1 + O(\alpha')),
\end{aligned}$$

as desired.

Moreover, we will apply this theorem with a protocol Π *which is padded by* $18\beta^{-1}e+20d\beta^{-1}e' = O(\alpha \cdot CC(\Pi'_{\mathcal{A}}))$ *zeros. Thus, Item 4 from Theorem 9 implies correctness.*

For example, one can set $\alpha = O(\min\{\sqrt{\epsilon}, (\epsilon')^{1/3}\})$*, and set* $\beta = O(\alpha)$*,* $d = \frac{1}{\alpha}$*,* $\alpha' = 1$*, to obtain an error resilient protocol in the ideal hash model with constant blowup in round complexity and* $1 + O(\alpha)$ *blowup in communication complexity.*

We defer the proof of Theorem 9 to Appendix B.

5 Hash Implementation with Shared Randomness

Recall that in Section 4, we presented an interactive coding scheme with the desired guarantees, in the ideal hash model, where we assume that the number of hash collisions is bounded, and where the budget for making a collision is proportional to the message length (where the message length is the maximum between the length of the message that was sent and the corrupted version of it). We denote this ideal protocol by Π.

In this section, we show how to implement the ideal hash with a real hash function. Loosely speaking, given a hash function h, we convert the protocol Π to the protocol Π^h which is identical to Π, where the ideal hash function is replaced by h. In order to maintain the desired efficiency and error-resilience guarantees, we need to ensure that, on the one hand, these hash values are not too long; and on the other hand there are not too many hash collisions (i.e., that these hashes form a good equality test). To ensure the latter condition holds, it is easy to see that we cannot use a single (deterministic) hash function. Instead we use a *family* of *randomized* hash functions.

We construct a function family $\mathcal{H} = \{h_x\}$, where each hash function h_x is associated with a (possibly long) seed x. In this section, we consider the *shared randomness model*, where the parties are allowed to share a (possibly long) random string. In Section 6 we show how to eliminate the need for shared randomness.

In this section we assume that the shared randomness is as long as we need. In particular, we use a different hash function (i.e, a different seed) for each equality query. Since the length of the protocol is adaptive and not a priori bounded[10], the length of the common random string is also not a priori bounded. We assume that there is a separate segment of the common random string for each round r, and each such segment contains five hash seeds, since in Π, in rounds that a party sends an ideal hash, the party sends five ideal hashes.

We emphasize that the shared randomness (and in particular the seeds) are known to the adversary. Therefore the adversary, given a seed x can try to skew the protocol and cause the parties to send many messages whose hashes collide. To get around this, we construct a hash family, where each h_x is a *randomized* hash function. When a party sends a hash of a value V, the party will choose randomness S and will send $(S, h_x(V, S))$. On the one hand, the randomness S needs to be short, since otherwise this will blowup the communication complexity by too much. On the other hand, the adversary cannot predict S, and thus will not be able to skew the messages of the parties towards ones which the hashes collide. We note that a similar idea of using a randomized hash function was used by Haeupler [Hae14], for the sake of improving the rate of his interactive coding scheme.

Before presenting our randomized hash family, we start with some preliminaries.

[10] In Section 6, we convert any such protocol in the *unbounded* shared randomness model into one that uses only private randomness.

5.1 Preliminaries

Chernoff bounds.

Lemma 11. *For any $N \in \mathbb{N}$, and any N independent Bernoulli random variables $X_1, \ldots, X_N$, each with mean $\leq \gamma$, it holds that*

$$\Pr[\sum_{i=1}^{N} X_i > 2\gamma N] \leq e^{-\frac{1}{3}\gamma N}.$$

Definition 12. *A distribution D over $\mathbb{F}_2^n$ is δ-*bias *if for any $v \in \mathbb{F}_2^n \setminus \{0^n\}$, we have that*

$$\left| \Pr_{x \sim D} \left[\sum_{i=1}^{n} v_i x_i = 0 \right] - \frac{1}{2} \right| \leq \delta \ .$$

Lemma 13. *[NN93] There exists an absolute constant $C \in \mathbb{N}$ and an efficiently computable function $G : \{0,1\}^* \to \{0,1\}^*$ such that for any size k and a uniformly random string $S \in \{0,1\}^{Ck}$, we have that $G(S) \in \{0,1\}^{2^k}$ is a 2^{-k}-biased distribution of length 2^k.*

Lemma 14 (6.3 from [Hae14])**.** *There exists a hash family $\mathcal{F} = \{\mathcal{F}_L\}_{L \in \mathbb{N}}$, such that for every $L \in \mathbb{N}$ it holds that $\mathcal{F}_L = \{f_x\}_{x \in \{0,1\}^{2L}}$, and for every $x \in \{0,1\}^{2L}$, $f_x : \{0,1\}^{\leq L} \to \{0,1\}$. Moreover, for any $k \in \mathbb{N}$ and for any vectors $V_1^A, ..., V_k^A, V_1^B, ..., V_k^B \in \{0,1\}^{\leq L}$ the following holds:*

1. *For uniform $x = (x_1, \ldots, x_k) \in (\{0,1\}^{2L})^k$ it holds that for each $i \in [k]$, the probability that $f_{x_i}(V_i^A) = f_{x_i}(V_i^B)$ is $\frac{1}{2}$ whenever $V_i^A \neq V_i^B$, and 1 whenever $V_i^A = V_i^B$. Moreover, for each $i \in [k]$ these probabilities are independent.*
2. *For δ-biased distribution $x = (x_1, \ldots, x_k) \in (\{0,1\}^{2L})^k$, it holds that the distribution*

$$\left(\mathbf{1}_{f_{x_1}(V_1^A) = f_{x_1}(V_1^B)}, \ldots, \mathbf{1}_{f_{x_k}(V_k^A) = f_{x_k}(V_k^B)} \right)$$

is δ-close to the case where x is uniform.

5.2 Our Hash Function

We are now ready to construct our family of randomized hash functions. We first define the randomized hash family $\mathcal{H}' = \{h'_x\}$, which uses the hash family $\mathcal{F}$ from Lemma 14. Recall that the hash values of $\mathcal{H}'$ should not be too long, since this will result in a large blowup in communication complexity.

In our construction, as opposed to previous constructions [BK12, BKN14, Hae14], the length of each hash value depends not only on the length of the message it is appended to, but it also depends on the length of the *entire* communication up until the point that the hash was sent. We note that if we were only concerned with the communication blowup and were not concerned with the round blowup, then we could have the length of the hash value depend only on the length of the message it is sent with (in similar spirit to prior work). However, as we argue below, in order to ensure a constant blowup in round complexity, we must allow the length of the hash value to also depend on the length of the entire history. This is illustrated in the following example: Suppose that a short message is sent, and prior to this short message were a few very long messages (in a way that satisfies the smoothness criterion). By corrupting a few long messages, the adversary can cause a hash collision in many short messages (with hash), which will result with a large blowup to the round complexity.

Hence, we allow the length of the hash value, not only to depend on the length of the message it is appended to, but also to depend on the length of the communication history. Note that the parties do not necessarily agree on the history length even if no errors occur in the current round, since the adversary may insert and delete bits throughout the protocol, in which case they will fail to parse the message and hash pair correctly.

Thus, we define the hash family $\mathcal{H} = \{h_x\}$, where the output of h_x includes the output of h'_x, the randomness used by h'_x (which is needed in order check for equality), and also the length of the hash value. Namely, we define

$$h_x(V) = (S, h'_x(V;S), 1 \cdot 0^w),$$

where S is the (private) randomness used by h'_x, and w is the length of $H'_x(V;S)$.

We next define h'_x. As we mentioned, the length of the hash values (denoted by w) may differ from one round to the next, as they depend on the communication complexity so far, and on the length of the current message sent. We will specify how w is defined below. But we first, define h'_x assuming w is known.

The seed x is random in $\left(\{0,1\}^{2L}\right)^L$, where we assume that L is greater than the input V (which is bounded by the communication complexity of the protocol up until the point where the hash is sent). For any $y = (y_1, \ldots, y_L) \in \left(\{0,1\}^{2L}\right)^L$ and for any $k \leq L$, let

$$f^k_y(V) = (f_{y_1}(V), ..., f_{y_k}(V)).$$

where $\mathcal{F} = \{f_y\}$ is the hash family from Lemma 14. The randomness for h'_x is denoted by S and is of size $2C \cdot w$. Let h'_x be the randomized hash function, that takes as input a variable V, randomness S, and outputs

$$h'_x(V;S) = f^w_{G(S)}(Z),$$

where

$$Z = \begin{cases} (f^{2^w}_x(V), 0) & \text{if } |V| \geq 2^w \\ (V, 1) & \text{if } |V| < 2^w \end{cases}$$

In what follows we show how the length w of the hash values are chosen. To this end we need to define the following variables with respect to a certain round r.

- Let Q^A be the set of all rounds r in which Alice send a hash to Bob.

  Note that these are exactly the set of rounds r such that r divides d or Alice send a message of length $\geq d$.

  We define Q^B analogously.

- Let a^A_r be the number of messages that Alice sent with a hash until (and including) round r. Namely,

  $$a^A_r = |\{r' \leq r : r' \in Q^A\}|.$$

  We define a^B_r analogously.

- Let t^A_r be all the communication received by Alice until (and including) round r. We define t^B_r analogously.

- For every r, if Alice sends the message in round r then define

  $$u_r = \max_{r' \in Q^A \cap [r-1]} \log \frac{t^A_r - t^A_{r'}}{a^A_r - a^A_{r'}}.$$

  The definition is analogous in the case that Bob sends the message in round r.

- For every round r, let ℓ_r denote the length of the message to be sent in round r, and let

$$w_r = \lceil \alpha \ell_r \rceil + \lceil u_r \rceil + 9 \left\lceil \log \frac{1}{\gamma} \right\rceil + 6,$$

where $\alpha, \gamma > 0$ are parameters of the scheme, where $\gamma < \alpha$ (it will be instructive to think of $\gamma = \epsilon$, where ϵ is the corruption budget of the adversary, and of α as the communication blowup in the error-resilient protocol).

5.3 Analysis

We denote by E the set of all messages $m_{A,r}$ or $m_{B,r}$ that were not corrupted but had a hash associated with them that formed a hash collision. Recall that for any set of messages T, we denote by $|T|$ the volume of T (i.e., the number of bits in T), and we denote by $|T|'$ the number of messages in T.

Lemma 15. *Fix $\epsilon < 0.0005$. The protocol $\Pi^{\mathcal{H}}$ defined in Section 5.2 satisfies the following: If $\Pi^{\mathcal{H}}$ consists of $\leq t$ bits and $\leq r$ rounds, then for any adversary that corrupts messages with total volume at most ϵt, we get that*

1. *With probability $\geq 1 - 10 \cdot e^{-\frac{\alpha\gamma}{3 \log \frac{1}{\gamma}} t}$ (over the common and private randomness),*

$$|E| \leq 20\gamma t.$$

2. *With probability $\geq 1 - 10e^{-\frac{2}{3}\gamma r}$ (over the common and private randomness),*

$$|E|' \leq 70\gamma r.$$

The proof of this Lemma is deferred to Appendix C.1.

5.4 Communication Bound

In this section we will bound the blowup of the communication of $\Pi^{\mathcal{H}}$, defined in Section 5.2. To this end, fix any adversary $\mathcal{A}$ for the protocol $\Pi^{\mathcal{H}}$, that corrupts at most e' messages of total volume at most e. We define a corresponding adversary $\mathcal{D}$ for the protocol Π, that corrupts at most e' message of total volume at most e, as follows:

The adversary $\mathcal{D}$ sends the exact same messages as $\mathcal{A}$ does, excluding the hash values. Recall that for each message in $\Pi^{\mathcal{H}}_{\mathcal{A}}$, the part that belongs to the hash value is well-defined by the suffix of the message $1 \cdot 0^w$, and hence the adversary $\mathcal{D}$ is well defined.

Lemma 16.

$$\mathrm{CC}(\Pi^{\mathcal{H}}_{\mathcal{A}}) \leq (1 + 50C\alpha)\,\mathrm{CC}(\Pi_{\mathcal{D}}) + e + 600C \log \frac{1}{\gamma} \cdot k$$

where C is the universal constant from Lemma 13, and k is the number of rounds with hash in $\Pi_{\mathcal{D}}$.

The proof of this lemma is deferred to Appendix C.2.

6 Hash Implementation with Private Randomness

In this section we show how to implement the ideal hashes in protocol Π, defined in Section 4, without resorting to shared randomness, but rather using only private randomness. To this end, we will slightly modify the protocol Π, into a new protocol Π'.

High-level overview of Π'. Lets first recall the approach used in previous works [BK12, Hae14]. In these works, the long shared randomness is replaced with δ-biased randomness, where $\delta = 2^{-\alpha t}$ and t is a bound on the communication complexity. Such δ-biased randomness can be generated using only $O(\alpha t)$ random bits. Hence, in previous works, these $O(\alpha t)$ bits of randomness are sent in advance (using an error correcting code). If we indeed had a bound t on the communication complexity, then this idea would work, as explained below.

Recall that the randomness is used for equality testing. From Lemma 14, we know that for any oblivious adversary (i.e., one that is independent of the randomness), the fraction of collisions in the case where the seed is random is δ-close to the fraction of collisions in the case where the seed is δ-biased. Denoting by N the number of possible oblivious adversaries, and by taking a union bound over all possible oblivious adversaries, we conclude that the probability that there exists an oblivious adversary that causes "too many" hash collisions in the case where the seed is δ-based is bounded by the same probability where the seed is truly random plus an additive term of δN. We note that

$$N \leq 2^{H(\epsilon)t} \cdot 4^{\epsilon t} = 2^{O(\epsilon \log \frac{1}{\epsilon} t)},$$

and thus

$$\delta N = 2^{-\alpha t} \cdot 2^{O(\epsilon \log \frac{1}{\epsilon} t)} = 2^{-\Omega(\alpha t)}.$$

Therefore, the probability that there exists an oblivious adversary that causes "too many" hash collisions is at most $2^{-\Omega(\alpha t)}$. Note that we can view any (non-oblivious) adversary as one that chooses an oblivious adversary as a function of the public randomness, and runs this oblivious adversary. Therefore, we conclude that for any (non-oblivious) adversary the probability that there are "too many" hash collisions is bounded by $2^{-\Omega(\alpha t)}$.

However, in our setting, we do not have an a priori bound on the communication complexity. In particular, if we replace the CRS with δ-biased randomness, where $\delta = 2^{-\alpha t}$ for some t, and if the adversary has a corruption budget of more than $O(\alpha t)$ bits (i.e., the communication complexity is larger than $\frac{\alpha t}{\epsilon}$), then our protocol is no longer safe. We overcome this problem by sending more randomness as the communication complexity increases.

More specifically, the parties start by assuming that the communication complexity is some small $t_{\min}$, where $t_{\min}$ is a lower bound on the communication complexity. So, the protocol starts when one of the paries, say Alice, chooses a random string $s \in \{0,1\}^{\alpha t_{\min}}$, and sends it to Bob.[11]

Once $t_1 \geq \frac{\alpha t_{\min}}{\epsilon}$ bits are sent in the protocol, the safety of this randomness could be compromised, since the adversary has enough budget to compromise $\alpha t_{\min}$ bits. Hence, each party, before sending its message, will check whether sending this message will cause the communication complexity to exceed $\frac{\alpha t_{\min}}{\epsilon}$. If so, then instead of sending the message, the party will send new randomness. This time, the party will choose at random s_2 of size $\alpha t_1 - |s_1|$ and send (s_1, s_2). If this randomness is inconsistent with the first randomness sent (s_1) then the party receiving the randomness aborts.

Once the communication complexity is $t_2 \geq \frac{\alpha t_1}{\epsilon}$, again the safety of the previous randomness could have been compromised, and hence as above, if a party is about to send a message that will cause the communication complexity to exceed $\frac{\alpha t_1}{\epsilon}$, then instead of sending the message, the party will choose at random s_3 such that $|s_1| + |s_2| + |s_3| = \alpha t_2$, and will send (s_1, s_2, s_3), etc. If at any point the randomness received is inconsistent with the previous random string then the party aborts. We refer to these special messages that transmit randomness by *system messages*.

[11] As before, we ignore the error-correcting code, since we consider only message adversaries, that corrupt messages as opposed to bits, and the budget for corrupting a message is the length of the message (or the length of the corrupted message, whichever is longer).

There is a slight problem with this idea: How does Bob know which message sent by Alice corresponds to a message in the initial protocol Π, and which is a system message? We fix this problem by appending 1's to system messages, and appending 0's to messages corresponding to Π. However, recall, that we do not want to blowup the communication complexity. Hence we only append these bits to long messages. This is enough, since system messages are always long.

Note that according to our protocol the parties first receive randomness s_1, then they receive new randomness (s_1, s_2), and so on. We ensure that if at any point, a system message was decoded incorrectly, then eventually the paries will abort, and "catch" the adversary with injecting too many errors. This guarantee simplifies the analysis: Either at some point a system message was decoded incorrectly, in which case the adversary is "caught" with injecting too many errors, or all the parties always agree on the randomness, in which case correctness follows from the correctness of the underlying protocol in the shared randomness model.

To ensure that indeed the parties will always notice when a system message was corrupted, we add to the system message the rounds $r_1, \ldots, r_k$ in which system messages were sent. This is done to circumvent the case where the message (s_1, s_2) was corrupted and converted into a protocol message, and a few rounds later a protocol message was corrupted and converted into the same system message (s_1, s_2). If we do not include the round number then the parties may never notice that there was a point in the protocol where they did not agree on the shared randomness. In order to avoid dealing with such cases, we simply include the round numbers of the system messages.

Finally, we notice that even though we ensure that the parties always agree on the shared randomness (assuming the adversary does not inject too many errors), there is still a subtle issue. Note that the first random string s_1 is δ-biased for $\delta = 2^{-\alpha t_{\min}}$. As we saw in previous work, this suffices if the number of oblivious adversaries, restricted to the first $t_{\min}$-bits, is bounded by $2^{O(\alpha t_{\min})}$. However, in our setting, since the total communication may be significantly larger than $t_{\min}$, the number of such oblivious adversaries can be as large as $2^{t_{\min}}$, in which case the number of rounds with hash collisions can be large. To overcome this problem, we ensure that in the first $t_{\min}$ bits of communication, the adversary cannot inject too many errors (without being "caught"). This is done by re-sending the first $t_{\min}$ bits after $t_{\min}/\alpha$ bits of the protocol were transmitted, and the parties abort if these $t_{\min}$ bits are ϵ/α-far from the first $t_{\min}$ bits of the transcript. More precisely, to each system message sent after t bits of the protocol were communicated, we append the first αt bits of the transcript.

In what follows we present our protocol Π'. For the sake of simplicity, after each system message is sent, the party receiving a system message replies with an "echo" message, by simply repeating the system message. The purpose of this "echo" message is simply to allow the other party to send his protocol message (which he didn't have the budget to send in the previous round).

The protocol Π'. Let $b > 2$, and let $\alpha \leq \frac{1}{3200C}$, where C is the constant defined in Lemma 13. Fix any $d \in \mathbb{N}$ and $\gamma > 0$ such that

$$\gamma \leq \min\left\{\frac{1}{d}, 2^{-b}\right\} \quad \text{and} \quad d \geq \frac{\log \frac{1}{\gamma}}{\alpha}. \tag{4}$$

For convenience the reader can think of $b = 2$ and $\gamma = \frac{1}{d}$.

Let Π be the protocol, in the ideal hash model, defined in Section 4, instantiated with α and d as above, and with any $\alpha' > 0$. Let $t_{\min}$ be a lower bound on the communication complexity of Π, where

$$t_{\min} \geq \max\{\alpha^{-2}, 250C\alpha^{-1} \log d\}, \tag{5}$$

and let

$$W = \alpha t_{\min}.$$

Let $\mathcal{H}$ be the hash family defined in Section 5. The protocol Π' makes oracle access to the protocol $\Pi^{\mathcal{H}}$ (defined in Section 5).

In protocol Π', each party maintains a transcript T initialized to $\emptyset$, an integer k initialized to 0, k strings $s_1, ..., s_k$, k partial transcripts $P_1, \ldots, P_k$, and k rounds $r_1, ..., r_k$ that will be determined during the protocol. Intuitively, T is the transcript corresponding to Protocol $\Pi^{\mathcal{H}}$, $s_1, \ldots, s_k$ are k seeds that are used to generate the hash function implementing the ideal hash, and $r_1, \ldots, r_k$ correspond to rounds in $\Pi^{\mathcal{H}}$ where the common randomness changes. Similarly to Π (and $\Pi^{\mathcal{H}}$), in Π' we interpret the (partial) transcripts as strings.[12] .

In Π', if a party aborts, it always waits until at least $t_{\min}$ bits are sent before aborting the protocol, so as to fulfill the requirement that the communication complexity of Π' is at least $t_{\min}$.

In the first round of Π' Alice does the following:

1. Choose $s_1 \in_R \{0,1\}^W$, and let $k = 1$, $r_1 = 0$, and $P_1 = \emptyset$.
2. Send $(s_1, P_1, r_1, 1)$.

We next describe the protocol from Alice's point of view, given her private state

$$(T, k, s_1, \ldots, s_k, r_1, \ldots, r_k, P_1, \ldots, P_k).$$

Bob's view is symmetric (by switching between A and B). Upon receiving a message m_B, Alice does the following

1. If in the previous round Alice computed her message in step 4(e)ii of the protocol (or if the previous round was the first round of the protocol) then check that m_B is an echo of (i.e., equal to) the message sent by Alice in the previous round. If not then halt, and otherwise goto Step 4c.
2. Otherwise, denote $\ell = |m_B|$.
3. If $\ell \geq b^k W$ and the least significant bit of m_B is 1, then do the following:
   (a) If there exists $s, P, r \in \{0,1\}^{b^k W}$, where r is a binary representation of $|T|'$, such that

   $$(s_1, ..., s_k, s, P_1, ..., P_k, P, r_1, ..., r_k, r, 1) = m_B,$$

   and such that P can be obtained from a prefix of T by corrupting messages of volume at most $\frac{|P|}{bd \log d}$, then define $s_{k+1} = s$, define $r_{k+1} = r$, $P_{k+1} = P$, update $k \leftarrow k + 1$, and send (an echo message) m_B.
   (b) Else, abort the protocol.
4. Else, do the following:
   (a) If $\ell \geq b^k W$ then let m'_B be the message m_B when the least significant bit of m_B is truncated. Otherwise, let $m'_B = m_B$.
   (b) Update $T \leftarrow T \cup \{m'_B\}$

[12] This is done by standard encoding, where after each bit of the transcript we add a bit that represents whether the message ended or not. Thus, a transcript of length ℓ can be described by a string of length 2ℓ.

(c) Define $m_A = \Pi^{\mathcal{H}}(T)$, using $x = x(s_1, \ldots, s_k, r_1, \ldots, r_k)$ as the shared randomness, where the exact function x is defined later (after Lemma 54). If there is no m_A to send then abort.

(d) If $|T \cup m_A| < \frac{b^k W}{400C\alpha}$ then do the following:

i. Update $T \leftarrow T \cup \{m_A\}$.

ii. Let $\ell = |m_A|$.

iii. If $\ell < b^k W$ then send m_A, and otherwise send $(m_A, 0)$.

(e) Else,

i. Let $s_{k+1}, P_{k+1}, r_{k+1} \in \{0,1\}^{b^k W}$ such that s_{k+1} is a uniformly chosen random string, P_{k+1} consists of the first $b^k W$ bits of T (where T is viewed as string) and r_{k+1} is a binary representation of $|T|'$.[13]

ii. Send
$$(s_1, \ldots, s_k, s_{k+1}, P_1, \ldots, P_k, P_{k+1}, r_1, \ldots, r_k, r_{k+1}, 1).$$

iii. $k \leftarrow k + 1$.

Theorem 17. *Fix any adversary $\mathcal{A}$ for Π' that corrupts $\epsilon' R(\Pi'_{\mathcal{A}})$ of the messages of total volume at most $\epsilon \mathrm{CC}(\Pi'_{\mathcal{A}})$, for $\epsilon \leq \frac{\alpha}{bd \log d}$, where $\Pi'_{\mathcal{A}}$ denotes the protocol Π' executed with the adversary $\mathcal{A}$. Then there exists an adversary $\mathcal{D}$ for the protocol Π, that corrupts at most $\epsilon' R(\Pi_{\mathcal{D}}) + 2\epsilon' \log_b \mathrm{CC}(\Pi_{\mathcal{D}})$ messages of total volume at most $2\epsilon \mathrm{CC}(\Pi_{\mathcal{D}})$,[14] such that the following holds:*

1. *$\Pi'_{\mathcal{A}}$ always sends at least $t_{\min}$ bits.*
2. *$\mathrm{CC}(\Pi'_{\mathcal{A}}) \leq (1 + 2600C\alpha)\,\mathrm{CC}(\Pi_{\mathcal{D}})$.*
3. *$R(\Pi'_{\mathcal{A}}) \leq R(\Pi_{\mathcal{D}}) + 2\log_b \mathrm{CC}(\Pi_{\mathcal{D}})$.*
4. *When $\Pi'_{\mathcal{A}}$ ends, both Alice and Bob (separately) can efficiently compute their view of the transcript of $\Pi_{\mathcal{D}}$.*
5. *The adversary $\mathcal{D}$ chooses the hash collisions in a probabilistic manner such that for every t and every r, with probability $\geq 1 - 20 \cdot 2^{-\frac{\gamma}{3d}t}$, the volume of hash collisions in the first t bits of $\Pi_{\mathcal{D}}$ is at most $35\gamma t$, and with probability $\geq 1 - 80r \cdot 2^{-\frac{7\gamma^8}{d}r}$, the number of rounds with hash collisions in the first r rounds of $\Pi_{\mathcal{D}}$ is at most $100\gamma r$.*
6. *Π' is efficiently computable if Π is efficiently computable.*

The proof of Theorem 17 is deferred to Appendix D.

7 Putting it all Together

In this section, we prove our main theorem (Theorem 4), using the theorems from previous sections. We restate our main theorem for the sake of convenience.

[13] The binary representation of $|T|'$ has length $\leq b^k W$ since $b^k W \geq \frac{1}{\alpha}$ and so $\log |T|' \leq \log \frac{b^k W}{\alpha} \leq b^k W$.

[14] $\Pi_{\mathcal{D}}$ denotes the protocol Π executed with the adversary $\mathcal{D}$.

Theorem 18. *There exists a universal constant $\alpha_0 \geq 0$ such that for any blowup parameters $\alpha \leq \alpha_0$ and $\alpha' \leq 1$, there exist parameters $\epsilon = \tilde{\Omega}\left(\alpha^{3+\frac{1}{\alpha'}}\right)$, $\epsilon' = \tilde{\Omega}(\alpha\alpha'^3)$, and $\delta = \alpha^{O(1/\alpha')}$, and there exists a probabilistic oracle machine S, such that for any protocol $\Pi = (A, B)$, in which the parties always transmit at least $t_{\min}$ bits (even in the presence of error), and for any adversary $\mathcal{A}$ that corrupts at most ϵ-fraction of the bits of the simulated protocol $\Pi' = (S^A, S^B)$, the protocol $\Pi'_{\mathcal{A}}$ (which is the protocol Π' executed with the adversary $\mathcal{A}$), satisfies the following properties.*

1. $\mathrm{CC}(\Pi'_{\mathcal{A}}) \geq t_{\min}$.
2. *There exists $t_0 = (1 + \tilde{O}(\alpha))\mathrm{CC}(A, B)$ such that for all $t > t_0$*
$$\Pr[\mathrm{CC}(\Pi'_{\mathcal{A}}) > t] \leq 2 \cdot 2^{-\delta t} ,$$
where the probability over the private randomness of S.
3. *There exists $r_0 = (1 + O(\alpha'))\, R(A, B) + O\left(\frac{1}{\log \frac{2}{\alpha'}} \log \mathrm{CC}(A, B) + 1\right)$ such that for any $r \geq r_0$, if at most ϵ'-fraction of the messages are α^2-corrupted, then*
$$\Pr\left[R(\Pi'_{\mathcal{A}}) > r\right] \leq 2 \cdot 2^{-\delta r} ,$$
where the probability over the private randomness of S.
4. *For any $t > 0$,*
$$\Pr\left[\left(\mathsf{Output}(\Pi'_{\mathcal{A}}) \neq \mathsf{Trans}(\Pi)\right) \ \wedge \ \left(\mathrm{CC}(\Pi'_{\mathcal{A}}) > t\right)\right] \leq 2 \cdot 2^{-\delta t} ,$$
where the probability over the private randomness of S.
5. *S is a probabilistic polynomial time oracle machine, and hence the computational efficiency of S^A and S^B is comparable to that of A and B, respectively.*

In the proof of this theorem, we use an error correcting code from a recent work of Guruswami and Li [GL16].

Theorem 19. *[GL16] For every $\alpha > 0$ there is an explicit encoding scheme $\mathrm{Enc}, \mathrm{Dec} : \{0,1\}^* \to \{0,1\}^*$ with the following properties:*

1. *For any $m \in \{0,1\}^*$ we have $|\mathrm{Enc}(m)| = (1 + \tilde{O}(\alpha))|m|$.*
2. *For any $m \in \{0,1\}^*$ and any y that can be obtained from $\mathrm{Enc}(m)$ by $\alpha^2 \cdot |\mathrm{Enc}(m)|$ insertions and deletions, $\mathrm{Dec}(y) = m$.*
3. Enc *and* Dec *are computable by a polynomial time Turing machine.*

In the proof of Theorem 18 we use the following padding claim.

Claim 20. *Let $\alpha, \beta \leq 0.1$, and $L_0 \geq \alpha^{-1}$. Then any $(\alpha, 2\beta)$-smooth protocol Π can be efficiently converted into an (α, β)-smooth protocol Π' such that*

- *Π can be computed from the first $(1 - 2\alpha)$ fraction of bits of Π'.*
- $\mathrm{CC}(\Pi) + L_0 \leq \mathrm{CC}(\Pi') \leq (1 + 13\alpha)\mathrm{CC}(\Pi) + 3L_0$.
- $R(\Pi') \leq R(\Pi) + \log_{\frac{1}{\beta}} \mathrm{CC}(\Pi) + \log_{\frac{1}{\beta}} L_0 + 1$.

The proof of this claim is deferred to Section E.1.

Proof of Theorem 18. Fix any $\alpha \leq \alpha_0$ and $\alpha' \leq 1$. Let C be the constant from Lemma 13 (see Section 5). Let $\mathrm{Enc}, \mathrm{Dec}$ be the encoding scheme from Theorem 19 with the parameter α. Recall that for all m, $|\mathrm{Enc}(m)| = (1+\tilde{O}(\alpha))|m|$. Let α_1 be the maximal constant that satisfies,

$$\forall m : |\mathrm{Enc}(m)| \leq 2|m|. \tag{6}$$

We define $\alpha_0 = \min\{\alpha_1, \frac{1}{3200C}\}$.

Given α, α' define,

$$\beta = \frac{\alpha^{\frac{1}{\alpha'}}}{320 \log^2 \frac{1}{\alpha}} \ , \ \gamma = \frac{\alpha^{\frac{4}{\alpha'}}}{2^{40}} \ , \ b = \frac{2}{\alpha'} \ , \ d = \frac{\log \frac{1}{\gamma}}{\alpha} \ , \ L_0 = 250C\alpha^{-2} \log d \ ,$$

$$\epsilon = \min\left\{\frac{\alpha^3}{2bd \log d}, \frac{\alpha^3 \beta}{320}\right\} \ , \ \epsilon' = \frac{\alpha \alpha'^3}{\log^3 \frac{1}{\alpha}}, \text{ and } \delta = \gamma^9 \ .$$

These parameters were chosen to satisfy the following claim.

Claim 21. *Our parameters satisfy the following:*[15]

1. $\epsilon = \tilde{\Omega}(\alpha^{3+\frac{1}{\alpha'}})$, $\epsilon' = \tilde{\Omega}(\alpha\alpha'^3)$ *and* $\delta = \alpha^{O(1/\alpha')}$.
2. $\alpha < \frac{1}{4}$ *and* $\beta \leq \frac{\alpha}{16}$.
3. $\alpha, \beta < 0.1$ *and* $L_0 \geq \alpha^{-1}$.
4. $\alpha \leq 0.01$, $\alpha' \leq 1$, $d \geq \frac{1}{\alpha}$ *and* $\beta \leq \min\left\{\alpha^{\frac{1}{\alpha'}}, \frac{1}{5\alpha d^2}\right\}$.
5. $\gamma \leq \min\left\{\frac{1}{d}, 2^{-b}\right\}$, $d \geq \frac{\log \frac{1}{\gamma}}{\alpha}$, *and* $L_0 \geq \max\{\alpha^{-2}, 250C\alpha^{-1} \log d\}$.
6. $\frac{2\epsilon}{\alpha^2} \leq \frac{\alpha}{bd \log d}$.
7. $18\beta^{-1}(35\gamma + \frac{4\epsilon}{\alpha^2}) + 20d\beta^{-1}(35\gamma + 4\epsilon) \leq \alpha$.
8. $(100\gamma + \epsilon') \cdot 906d \log \frac{1}{\beta} \leq \alpha'$ *and* $1812d \log \frac{1}{\beta}\epsilon' \leq \frac{1}{\log \frac{2}{\alpha'}}$.
9. $\delta \leq \frac{1}{(10\alpha^{-1}+10)L_0}$, *and for all* $x > 0$ *we have that*

$$2 \cdot 2^{-\delta x} \geq \min\left\{1, \frac{120d}{\gamma} \cdot 2^{-\frac{\gamma}{3d}x} + \gamma^{-17} \cdot 2^{-\frac{3}{2}\gamma^8 x}\right\} \ .$$

The protocol Π' is defined as follows:

1. Convert Π into a $(\alpha, 2\beta)$-smooth protocol Π_{smooth} by applying Lemma 6 (Section 3) to Π, with respect to parameters $(\alpha, 2\beta)$. These parameters satisfy the requirements in Lemma 6 by Item 2.
2. Convert Π_{smooth} into Π_{pad} using Claim 20 (above) with parameters α, β, L_0. These parameters satisfy the requirements in Claim 20 by Item 3.

[15] Each of the following items will later be used to apply a different theorem from previous sections.

3. Convert Π_{pad} to the error-resilient protocol Π_{ideal}, which is error-resilient in the ideal hash model, by applying the protocol from Section 4 to Π_{pad}, with parameters $\alpha, \alpha', \beta, d$. Jumping ahead, note that by Item 4, these parameters satisfy Equation (3) which is required in order to apply Theorem 9.

4. Convert Π_{ideal} to the protocol Π_{rand}, which is obtained by instantiating the ideal hash using private randomness, obtained by applying the protocol described in Section 6 to Π_{ideal}. Jumping ahead, note that by Items 5 and 6, imply that Equations (4) and (5) are satisfied and the requirements of Theorem 17 are satisfied with respect to any adversary $\mathcal{A}$ that corrupts messages of total volume $\leq \frac{2\epsilon}{\alpha^2}\mathrm{CC}((\Pi_{\mathsf{rand}})_{\mathcal{A}})$.

5. Convert Π_{rand} to $\Pi' = (S^A, S^B)$, where Π' is the same as Π_{rand}, except that each message is sent encoded with the error correcting code from Theorem 19 with parameter α.

Lemma 22. *The protocol $\Pi' = (S^A, S^B)$ satisfies the conditions of Theorem 18.*

The proof of Lemma 22 is deferred to Appendix E.3.

□

References

[AGS13] Shweta Agrawal, Ran Gelles, and Amit Sahai. Adaptive protocols for interactive communication. Manuscript, arXiv:1312.4182 (cs.DS), 2013.

[AGS16] Shweta Agrawal, Ran Gelles, and Amit Sahai. Adaptive protocols for interactive communication. In *IEEE International Symposium on Information Theory, ISIT 2016, Barcelona, Spain, July 10-15, 2016*, pages 595–599, 2016.

[BE14] Mark Braverman and Klim Efremenko. List and unique coding for interactive communication in the presence of adversarial noise. In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, FOCS '14, pages 236–245, 2014.

[BEGH16] Mark Braverman, Klim Efremenko, Ran Gelles, and Bernhard Haeupler. Constant-rate coding for multiparty interactive communication is impossible. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 999–1010, 2016.

[BGMO16] Mark Braverman, Ran Gelles, Jieming Mao, and Rafail Ostrovsky. Coding for interactive communication correcting insertions and deletions. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 61:1–61:14, 2016.

[BK12] Zvika Brakerski and Yael Tauman Kalai. Efficient interactive coding against adversarial noise. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 160–166, 2012.

[BKN14] Zvika Brakerski, Yael Tauman Kalai, and Moni Naor. Fast interactive coding against adversarial noise. *J. ACM*, 61(6):35:1–35:30, 2014.

[BN13] Zvika Brakerski and Moni Naor. Fast algorithms for interactive coding. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '13, pages 443–456, 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*, STOC '11, pages 159–166, New York, NY, USA, 2011. ACM.

[Bra12] Mark Braverman. Towards deterministic tree code constructions. In *Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012*, pages 161–167, 2012.

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

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

[GH13] Mohsen Ghaffari and Bernhard Haeupler. Optimal error rates for interactive coding II: efficiency and list decoding. *CoRR*, abs/1312.1763, 2013.

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

[GHK+16] Ran Gelles, Bernhard Haeupler, Gillat Kol, Noga Ron-Zewi, and Avi Wigderson. Towards optimal deterministic coding for interactive communication. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1922–1936, 2016.

[GHS14] Mohsen Ghaffari, Bernhard Haeupler, and Madhu Sudan. Optimal error rates for interactive coding I: adaptivity and other settings. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 794–803, 2014.

[GK17] Ran Gelles and Yael Tauman Kalai. Constant-rate interactive coding is impossible, even in constant-degree networks. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:95, 2017.

[GL16] Venkatesan Guruswami and Ray Li. Efficiently decodable insertion/deletion codes for high-noise and high-rate regimes. In *IEEE International Symposium on Information Theory, ISIT 2016, Barcelona, Spain, July 10-15, 2016*, pages 620–624, 2016.

[GMS11] Ran Gelles, Ankur Moitra, and Amit Sahai. Efficient and explicit coding for interactive communication. In *Proceeding of the IEEE Symposium on Foundations of Computer Science*, FOCS '11, pages 768–777, 2011.

[Hae14] Bernhard Haeupler. Interactive Channel Capacity Revisited. In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, FOCS '14, pages 226–235, 2014.

[HS17] Bernhard Haeupler and Amirbehshad Shahrasbi. Synchronization strings: codes for insertions and deletions approaching the singleton bound. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 33–46, 2017.

[HSV17] Bernhard Haeupler, Amirbehshad Shahrasbi, and Ellen Vitercik. Synchronization strings: Channel simulations and interactive coding for insertions and deletions. *CoRR*, abs/1707.04233, 2017.

[JKL15] Abhishek Jain, Yael Tauman Kalai, and Allison Lewko. Interactive coding for multiparty protocols. In *Proceedings of the 6th Conference on Innovations in Theoretical Computer Science, ITCS '15*, pages 1–10, 2015.

[KR13] Gillat Kol and Ran Raz. Interactive channel capacity. In *STOC '13: Proceedings of the 45th annual ACM symposium on Symposium on theory of computing*, pages 715–724, New York, NY, USA, 2013. ACM.

[NN93] Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993.

[RS94] Sridhar Rajagopalan and Leonard Schulman. A coding theorem for distributed computation. In *STOC '94: Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 790–799, New York, NY, USA, 1994. ACM.

[Sch92] Leonard J. Schulman. Communication on noisy channels: a coding theorem for computation. *Foundations of Computer Science, Annual IEEE Symposium on*, pages 724–733, 1992.

[Sch93] Leonard J. Schulman. Deterministic coding for interactive communication. In *STOC '93: Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, pages 747–756, New York, NY, USA, 1993. ACM.

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

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

[SW17] Alexander A. Sherstov and Pei Wu. Optimal interactive coding for insertions, deletions, and substitutions. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:79, 2017.

A Smooth Protocols

In this section we prove Lemma 6. Namely, we show how to convert any protocol into a smooth protocol. Recall the definition of a smooth protocol.

Definition 23. *A protocol is (α, β)-smooth if for every round r the following holds:*

$$\alpha \cdot \max\{|M_{r-1}|, |M_{r-2}|, |M_{r-3}|\} \leq |M_r| \leq \frac{1}{\beta} \cdot \min\{|M_{r-1}|, |M_{r-2}|, |M_{r-3}|\} \tag{7}$$

Recall Lemma 6.

Lemma 6. *For any $\alpha < \frac{1}{4}$ and $\beta \leq \frac{\alpha}{8}$, the following holds: Any protocol Π can be efficiently converted into an (α, β)-smooth protocol S^{Π} such that*

1. $\mathrm{CC}(S^{\Pi}) \leq \mathrm{CC}(\Pi) \cdot (1 + 50\alpha)$.
2. $R(S^{\Pi}) \leq R(\Pi) \cdot (1 + 8\log_{2\beta} \alpha) + 4\log_{\frac{1}{2\beta}} \cdot \mathrm{CC}(\Pi) + 4$
3. *If Π is computationally efficient then so is S^{Π}.*

Proof of Lemma 6. We denote the messages corresponding to the underlying protocol Π by $m_1, m_2, \ldots$, where m_t corresponds to the t'th round message of Π. We denote the messages corresponding to the smooth protocol S^{Π} by $M_1, M_2, \ldots$, where M_r corresponds to the r'th round message of S^{Π}. In what follows, we describe the protocol from the side of Alice in the $(r+1)^{\text{st}}$ round of the protocol S^{Π}, after receiving a message M_r from Bob.

Suppose that, before receiving this message, Alice has recovered all the messages corresponding to the first $t-1$ rounds of Π (and possibly a prefix of the t'th round message). We denote by T the (partial) transcript that Alice holds. Formally, T is defined inductively starting with $T = \emptyset$, as follows.

1. Let
$$d_r = \lfloor \alpha \cdot \max\{|M_{r-1}|, |M_{r-2}|, |M_{r-3}|\} \rfloor$$
and let
$$k'_r = \left\lfloor \frac{1}{\beta} \cdot \min\{|M_{r-1}|, |M_{r-2}|, |M_{r-3}|\} \right\rfloor .$$

2. If $|M_r| = k'_r$, then parse $M_r = m'_t \cdot 0 \cdot 1^p$, update $T \leftarrow (T, m'_t)$, and send back the message $M_{r+1} = 0^p$.

   A message of length k'_r sent by Bob, is always interpreted as Bob not being done sending his message due to budget constraints. In this case, think of m'_t as being part of m_t, the t'th round message of Π (if it is the first part of m_t then it is a prefix, and if it not the first part, then it is the prefix of the remaining part of m_t). The length of the acknowledgment p is dictated by Bob, based on the length of the actual message that he is trying to send.

   Following such a message (of budget request), there will be three messages of the form $M_{r+1} = M_{r+2} = M_{r+3} = 0^p$.

3. Otherwise, $|M_r| < k'_r$. We distinguish between three cases:

   (a) Case 1: $M_r = 0^p$ and in the previous round, Alice sent a message of length k'_{r-1} of the form $M_{r-1} = m' \cdot 0 \cdot 1^p$. In this case send $M_{r+1} = 0^p$.

   This corresponds to the case that in the previous round Alice requested for more budget, since she did not have enough budget to finish sending her message m_{t-1}, Hence, Bob replied to her request with sending a "budget message" 0^p. In this case Alice and Bob each send another budget message, to ensure that in the next time Alice speaks she has the budget she requested.

   (b) Case 2: $M_r = M_{r-1} = 0^p$ and M_{r-2}, which was sent by Bob, is a message of length k'_{r-2}. In this case send $M_{r+1} = 0^p$.

   This corresponds to the case that in round $r-2$ Bob requested for more budget, since he did have enough budget to finish sending his message m_t. As mentioned above, after such a request, three "budget messages" of form 0^p are sent.

(c) Case 3: $M_r = M_{r-1} = M_{r-2} = 0^p$ and in round $r-3$ Alice sent a message of the form $M_{r-3} = m' \cdot 0 \cdot 1^p$ of length k'_{r-3}. As before, this corresponds to the case that Alice did not have enough budget to finish sending her message m_{t-1}. Namely, she has been wanting to send m_{t-1} but so far due to budget constraints she has only sent m'_{t-1} which is a prefix of m_{t-1}. However, in this case the three previous messages where of lengths p, and thus now Alice has a budget of $\left\lfloor \frac{p}{\beta} \right\rfloor$.

In this case, let m''_{t-1} be the remaining (suffix) of M_{t-1}; i.e.,

$$m_{t-1} = (m''_{t-1}, m'_{t-1}).$$

In what follows, we denote m''_{t-1} by m.

(d) Case 4: Otherwise, parse $M_r = m_t \cdot 0^{d_r}$. In this case, update $T \leftarrow (T, m_t)$, let m_{t+1} be the next message that Alice is supposed to send according to the updated transcript T, and denote m_{t+1} by m.

Note that this corresponds to the case that Alice had enough budget to finish sending m_{t-1} in the previous round, and Bob has enough budget to send all of m_t in the r'th round of Π'.

4. Compute

$$d_{r+1} = \lfloor \alpha \cdot \max\{|M_r|, |M_{r-1}|, |M_{r-2}|\} \rfloor ,$$

and

$$k'_{r+1} = \left\lfloor \frac{1}{\beta} \cdot \min\{|M_r|, |M_{r-1}|, |M_{r-2}|\} \right\rfloor .$$

Intuitively, Alice would like to send m with a padding of d_{r+1} zeros, in order to ensure that the condition $|M_{r+1}| \geq \alpha \cdot \max\{|M_r|, |M_{r-1}|, |M_{r-2}|\}$ is satisfied. However, we need to make sure that we do not violate the condition that $|M_{r+1}| \leq \frac{1}{\beta} \cdot \min\{|M_r|, |M_{r-1}|, |M_{r-2}|\}$. If this condition is violated then we do not send the (padded) message all at once, but rather we do this in phases, as described below.

5. If $|m| + d_{r+1} < k'_{r+1}$ then send $M_{r+1} \triangleq m \cdot 0^{d_{r+1}}$, update $T = (T, m)$, and halt.

6. Else, send $M_{r+1} \triangleq m' \cdot 0 \cdot 1^p$, where $p = \min\{k'_{r+1} - 2, \lceil 2\alpha|m| \rceil\}$ and m' is the prefix of m of length $k'_{r+1} - p - 1$. Update $T = (T, m')$, and halt.

   For Step 6 to be well defined, we must prove the following claim (whose proof is deferred to the sequel).

   Claim 24. *For every $r \in \mathbb{N}$, if $|m| + d_{r+1} \geq k'_{r+1}$ then $|m| > k'_{r+1} - p - 1$, where $p = \min\{k'_{r+1} - 2, \lceil 2\alpha|m| \rceil\}$.*

   This message M_{r+1} will be interpreted by Bob as saying that Alice would like to send a long message but does not have the budget to do so. m' will be interpreted as a prefix of Alice's message, and 1^p indicates that Alice wants the next three messages to be $M_{r+2} = M_{r+3} = M_{r+4} = 0^p$, which gives her the budget she needs to continue to send her message.

Correctness. We prove by induction on r that at the beginning of the r^{th} round of S^{Π}, the transcripts of both Alice and Bob, denoted by T_A and T_B respectively, are always a prefix of the original transcript (where the last message in T_A and T_B may be a prefix of a message sent in T).

For the base case, note that for $r = 1$, it holds that $T_A = T_B = \emptyset$. For the induction step, suppose the hypothesis is true for every round $< r$, and we will prove that the hypothesis is true for round r. We prove the correctness for T_A, a similar argument can be used to prove the correctness of T_B.

We first note that Alice can distinguish between whether the conditions of Step 2, 3a, 3b, 3c or 3d hold. This is the case, since in Step 2 the condition is that $|M_r| = k'_r$, whereas in Steps 3a, Step 3b, 3c and 3d one of the conditions is that $|M_r| < k'_r$, and note that Alice can compute k'_r on her own. In addition, if $|M_r| < k'_r$, she can distinguish between whether the conditions of Step 3a, 3b, 3c or 3d are satisfied, since she can compute k'_{r-1}, k'_{r-2} and k'_{r-3} on her own. Therefore, she knows whether she should answer with a "budget message".

If Alice adds the message m_t to T_A in Step 3d, then she received $M_r = m_t \cdot 0^{d_r}$ of length less than k'_r from Bob, where $|m_t| \geq 1$. Note that Alice can compute d_r on her own, and hence can compute m_t from M_r correctly.

If Alice adds the message m'_t to T_A in Step 2, then she received $M_r = m'_t \cdot 0 \cdot 1^p$ of length exactly k'_r from Bob. In this case, Alice can decode M_r and find m'_t by simply deleting from M_r the suffix of the form $0 \cdot 1^p$. She interprets m'_t as a prefix of the message m_t, or a prefix of the remaining m_t (note that a prefix of m_t could have already been sent in previous rounds).

By our induction hypothesis, T_A is a correct prefix of the original transcript. Hence, after Alice updates $T_A = (T_A, m_t)$ or $T_A = (T_A, m'_t)$, T_A remains a prefix of the transcript of Π.

We note that in Steps 3a, 3b, and 3c Alice does not update T. Indeed in these cases she shouldn't update T, since thesse correspond to the cases that in the previous round she received a "budget message".

We next prove that the (α, β) smoothness condition holds. The proof will be by induction on the number of rounds. However, we first prove the following claim, which will be used in the remaining of the proof.

Claim 25. *Fix any $r \in \mathbb{N}$, and suppose the (α, β) smoothness condition holds for all rounds $\leq r$. Then*

$$|k'_r| \geq \left\lfloor \frac{d_r}{\alpha} \right\rfloor$$

Proof. Fix any $r \in \mathbb{N}$. Recall that by definition,

$$k'_r = \left\lfloor \frac{1}{\beta} \cdot \min\{|M_{r-1}|, |M_{r-2}|, |M_{r-3}|\} \right\rfloor$$

and

$$d_r = \lfloor \alpha \cdot \max\{|M_{r-1}|, |M_{r-2}|, |M_{r-3}|\} \rfloor .$$

Let $c \in \{1, 2, 3\}$ be such that

$$|M_{r-c}| = \min\{|M_{r-1}|, |M_{r-2}|, |M_{r-3}|\}.$$

The (α, β)-smoothness of S^{Π} up until round r, together with the assumption that $\beta \leq \alpha$ implies that

$$|M_{r-c}| \geq \beta \cdot \max\{|M_{r-1}|, |M_{r-2}|, |M_{r-3}|\}.$$

Therefore,

$$k'_r = \left\lfloor \frac{|M_{r-c}|}{\beta} \right\rfloor \geq \max\{|M_{r-1}|, |M_{r-2}|, |M_{r-3}|\} \geq \frac{d_r}{\alpha} \geq \left\lfloor \frac{d_r}{\alpha} \right\rfloor ,$$

as desired.

□

The following corollary follows immediately from Claim 25, together with the observation that if $d_{r+1} = 0$ then $k'_{r+1} \geq \left\lfloor \frac{1}{\beta} \right\rfloor \geq 2$.

Corollary 26. *Fix any $r \in \mathbb{N}$, and suppose the (α, β) smoothness condition holds for all rounds $\leq r$. Then*

$$|k'_r| \geq |d_r| + 2$$

Smoothness. We now prove that the protocol S^{Π} is (α, β)-smooth. The proof is by induction on r. The base case is trivial. For the induction step, we first prove α-smoothness. Specifically, we prove that

$$|M_r| > d_r = \lfloor \alpha \cdot \max\{|M_{r-1}|, |M_{r-2}|, |M_{r-3}|\} \rfloor .$$

We distinguish between three cases:

1. The message M_r was sent in Step 5. Recall that in this case, $|M_r| > d_r$, as desired.
2. The message M_r was sent in Step 6. Recall that in this case, $|M_r| = k'_r > d_r$, as desired.
3. The message M_r was sent in Step 2, 3a or 3b. Recall that in these cases

   $$|M_r| = p = \min\{k'_{r-c} - 2, \lceil 2\alpha |m| \rceil\},$$

   for some $c \in \{1, 2, 3\}$, and where m is the message that Bob or Alice were trying to send in round $r - c$, but did not have enough budget to do so. In this case $|M_{r-c}| = k'_{r-c}$. We next prove the following claim

   Claim 27. $d_r = \lfloor \alpha \cdot k'_{r-c} \rfloor$.

   Proof. We distinguish between three cases:

   Case 1: $c = 1$. In this case,

   $$|M_{r-1}| = k'_{r-1} = \left\lfloor \frac{1}{\beta} \cdot \min\{|M_{r-2}|, |M_{r-3}|, |M_{r-4}|\} \right\rfloor .$$

   Note that in this case in order to prove Claim 27, it suffices to prove that

   $$|M_{r-2}|, |M_{r-3}| \leq k'_{r-1}.$$

   We start by proving that $|M_{r-2}| \leq k'_{r-1}$. To this end, note that by our induction hypothesis,

   $$|M_{r-2}| \leq k'_{r-2} = \left\lfloor \frac{1}{\beta} \cdot \min\{|M_{r-3}|, |M_{r-4}|, |M_{r-5}|\} \right\rfloor .$$

   Note that if $|M_{r-2}| \geq \min\{|M_{r-3}|, |M_{r-4}|\}$ then indeed $k'_{r-1} \geq |M_{r-2}|$. On the other hand, if $|M_{r-2}| < \min\{|M_{r-3}|, |M_{r-4}|\}$ then $k'_{r-1} = \left\lfloor \frac{1}{\beta} \cdot |M_{r-2}| \right\rfloor > |M_{r-2}|$. Thus, we conclude that $|M_{r-2}| \leq k'_{r-1}$, as desired.

   We next argue that $|M_{r-3}| \leq k'_{r-1}$. To this end, by our induction hypothesis,

   $$|M_{r-3}| \leq k'_{r-3} = \left\lfloor \frac{1}{\beta} \cdot \min\{|M_{r-4}|, |M_{r-5}|, |M_{r-6}|\} \right\rfloor .$$

If $\min\{|M_{r-2}|, |M_{r-3}|\} \geq |M_{r-4}|$ then $|M_{r-3}| \leq k'_{r-1}$, as desired. On the other hand, if $\min\{|M_{r-2}|, |M_{r-3}|\} < |M_{r-4}|$ then

$$k'_{r-1} = \left\lfloor \frac{1}{\beta} \cdot \min\{|M_{r-2}|, |M_{r-3}|\} \right\rfloor > \frac{1}{\beta} \cdot \min\{|M_{r-2}|, |M_{r-3}|\} - 1,$$

and hence

$$\min\{|M_{r-2}|, |M_{r-3}|\} < \beta \cdot (k'_{r-1} + 1).$$

Therefore, either $|M_{r-3}| < \beta \cdot (k'_{r-1} + 1) < k'_{r-1}$, as desired, or $|M_{r-2}| < \beta \cdot (k'_{r-1} + 1)$, which in turn implies that

$$\beta \cdot (k'_{r-1} + 1) > |M_{r-2}| \geq d_{r-2} = \lfloor \alpha \cdot \max\{|M_{r-3}|, |M_{r-4}|, |M_{r-5}|\} \rfloor \geq \lfloor \alpha \cdot |M_{r-3}| \rfloor ,$$

which in turn implies that

$$|M_{r-3}| \leq \frac{1}{\alpha} \left(\beta \cdot \left(k'_{r-1} + 1 \right) + 1 \right) = \frac{\beta}{\alpha} \cdot k'_{r-1} + \frac{\beta}{\alpha} + \frac{1}{\alpha} \leq k'_{r-1}$$

as desired, where the latter follows from the fact that $k'_{r-1} \leq \frac{1}{\beta}$ together with the fact that $\beta \leq \frac{\alpha}{8}$.

Case 3: $c = 3$**.** In this case, by definition, $|M_{r-2}| = |M_{r-1}| = p < k'_{r-3}$, and hence

$$d_r = \lfloor \alpha \cdot \max\{|M_{r-1}|, |M_{r-2}|, |M_{r-3}|\} \rfloor = \lfloor \alpha \cdot k'_{r-3} \rfloor,$$

as desired.

Case 2: $c = 2$**.** In this case, by definition, $|M_{r-1}| = p < k'_{r-2}$. Thus, to prove Claim 27 it suffices to prove that $|M_{r-3}| \leq k'_{r-2}$.

To this end, note that by the induction hypothesis, M_{r-3} satisfies the (α, β)-smoothness requirement, and hence

$$|M_{r-3}| \leq k'_{r-3} = \left\lfloor \frac{1}{\beta} \cdot \min\{|M_{r-4}|, |M_{r-5}|, |M_{r-6}|\} \right\rfloor .$$

Moreover, recall that by definition,

$$k'_{r-2} = \left\lfloor \frac{1}{\beta} \cdot \min\{|M_{r-3}|, |M_{r-4}|, |M_{r-5}|\} \right\rfloor .$$

Thus, if $|M_{r-3}| \geq \min\{|M_{r-4}|, |M_{r-5}|\}$ then indeed $|M_{r-3}| \leq k'_{r-2}$. On the other hand, if $|M_{r-3}| < \min\{|M_{r-4}|, |M_{r-5}|\}$ then $k'_{r-2} = \left\lfloor \frac{1}{\beta} \cdot |M_{r-3}| \right\rfloor > |M_{r-3}|$, as desired.

Therefore, we conclude that $|M_{r-3}| \leq k'_{r-2}$, and hence

$$d_r = \lfloor \alpha \cdot \max\{|M_{r-1}|, |M_{r-2}|, |M_{r-3}|\} \rfloor = \lfloor \alpha \cdot \max\{p, k'_{r-2}\} \rfloor = \lfloor \alpha \cdot k'_{r-2} \rfloor,$$

as desired. □

Armed with Claim 27, it is now easy to see that $k'_{r-c} - 2 > d_r$ since according to our assumptions, $\alpha \leq \frac{1}{4}$ and hence

$$k'_{r-c} - d_r = k'_r - \lfloor \alpha \cdot k'_r \rfloor \geq \frac{3k'_r}{4} \geq \frac{3\left\lfloor \frac{1}{\beta} \right\rfloor}{4} > 2$$

where the latter follows from our assumption that $\beta \leq \frac{1}{8}$.

Thus, it remains to argue that

$$\lceil 2\alpha|m| \rceil \geq d_r = \lfloor \alpha \cdot k'_{r-c} \rfloor,$$

which in turn implies that it suffices to argue that

$$2|m| \geq k'_{r-c}.$$

To this end, note that in this case

$$|m| + d_{r-c} \geq k'_{r-c},$$

and hence

$$\begin{aligned}
&2|m| \geq \\
&2(k'_{r-c} - d_{r-c}) = \\
&k'_{r-c} + (k'_{r-c} - 2d_{r-c}) \geq \\
&k'_{r-c} + (k'_{r-c} - 2\alpha(k'_{r-c} + 1)) = \\
&k'_{r-c} + (1 - 2\alpha)k'_{r-c} - 2\alpha \geq \\
&k_{r-c},
\end{aligned}$$

as desired, where the third equation follows from Claim 25 (together with induction hypothesis), and the last equation follows from our assumption that $\alpha \leq \frac{1}{4}$ and $\beta \leq \frac{1}{8}$.

We next prove β-smoothness. As above, we distinguish between three cases:

1. The message M_r was sent in Step 5. Recall that in this case, $|M_r| < k'_r$, as desired.
2. The message M_r was sent in Step 6. Recall that in this case, $|M_r| = k'_r$, as desired.
3. The message M_r was sent in Step 2 or 3a or 3b. Recall that in this case there exists $c \in \{1, 2, 3\}$ such that

$$|M_r| = p = \min\{k'_{r-c} - 2, \lceil 2\alpha|m| \rceil\} < k'_{r-c}.$$

Thus, in order to finish the proof of β-smoothness, it remains to prove the following claim.

Claim 28. $k'_{r-c} \leq k'_r$

Proof. We distinguish between three cases.

Case 1: $c = 1$**.** In this case, $|M_{r-1}| = k'_{r-1}$. This, together with our induction hypothesis, implies that for every $i \in \{2, 3, 4\}$,

$$|M_{r-1}| = k'_{r-1} = \left\lfloor \frac{1}{\beta} \min\{|M_{r-2}|, |M_{r-3}|, |M_{r-4}|\} \right\rfloor \leq \frac{1}{\beta} \cdot |M_{r-i}|.$$

Thus, for every $i \in \{2, 3, 4\}$,

$$|M_{r-i}| \geq \beta \cdot k'_{r-1}.$$

Hence,

$$k'_r = \left\lfloor \frac{1}{\beta} \cdot \min\{|M_{r-1}|, M_{r-2}|, |M_{r-3}|\} \right\rfloor \geq \left\lfloor \frac{1}{\beta} \cdot \beta \cdot k'_{r-1} \right\rfloor \geq k'_{r-1},$$

as desired.

Case 2: $c = 2$**.** In this case, $|M_{r-2}| = k'_{r-2}$ and $|M_{r-1}| = p < k'_{r-2}$. Moreover, in this case,

$$|M_{r-2}| = k'_{r-2} = \left\lfloor \frac{1}{\beta} \min\{|M_{r-3}|, |M_{r-4}|, |M_{r-5}|\} \right\rfloor \leq \frac{1}{\beta} \cdot |M_{r-3}|,$$

and hence

$$|M_{r-3}| \geq \beta \cdot k'_{r-2}.$$

Thus,

$$k'_r = \left\lfloor \frac{1}{\beta} \cdot \min\{|M_{r-1}|, M_{r-2}|, |M_{r-3}|\} \right\rfloor \geq \left\lfloor \frac{1}{\beta} \cdot \min\{p, k'_{r-2}, \beta \cdot k'_{r-2}\} \right\rfloor.$$

Therefore, in this case, in order to prove that $k'_r \geq k'_{r-c}$ it suffices to prove that

$$\left\lfloor \frac{p}{\beta} \right\rfloor \geq k'_{r-c}. \tag{8}$$

Recall that

$$p = \min\{k'_{r-c} - 2, \lceil 2\alpha|m| \rceil\}.$$

If $p = k'_{r-c} - 2$ then Equation (8) clearly holds since in this case

$$\left\lfloor \frac{p}{\beta} \right\rfloor = \left\lfloor \frac{k'_{r-c} - 2}{\beta} \right\rfloor > k'_{r-c}.$$

If $p = \lceil 2\alpha|m| \rceil$ then

$$\left\lfloor \frac{p}{\beta} \right\rfloor = \left\lfloor \frac{\lceil 2\alpha|m| \rceil}{\beta} \right\rfloor \geq \left\lfloor \frac{\lceil 2\alpha \cdot (k'_{r-c} - d_{r-c}) \rceil}{\beta} \right\rfloor \geq \left\lfloor \frac{\lceil 2\alpha \cdot (1 - \alpha) \cdot k'_{r-c} - 1 \rceil}{\beta} \right\rfloor \geq k'_{r-c},$$

as desired.

Case 3: $c = 3$**.** In this case, $|M_{r-3}| = k'_{r-3}$ and $|M_{r-2}| = |M_{r-1}| = p$. Thus,

$$k'_r = \left\lfloor \frac{1}{\beta} \cdot \min\{|M_{r-1}|, M_{r-2}|, |M_{r-3}|\} \right\rfloor = \left\lfloor \frac{1}{\beta} \cdot \min\{p, k'_{r-c}\} \right\rfloor \geq k'_{r-c},$$

as desired, where the latter follows from Equation (8). □

Now that we proved that the protocol S^{Π} is (α, β)-smooth, we next use Claim 25 in order to prove Claim 24.

Proof of Claim 24. Fix any $r \in \mathbb{N}$ such that $|m| + d_{r+1} \geq k'_{r+1}$. We argue that in this case, $|m| > k'_{r+1} - p - 1$, where $p = \min\{k'_{r+1} - 2, \lceil 2\alpha|m| \rceil\}$. Note that it suffices to prove that $p \geq d_{r+1}$. We distinguish between two cases:

Case 1: $\lceil 2\alpha|m| \rceil \geq k'_{r+1} - 2$**.** In this case, $p = k'_{r+1} - 2 \geq d_{r+1}$, where the latter inequality follows from Corollary 26.

Case 2: $\lceil 2\alpha|m| \rceil < k'_{r+1} - 2$**.** In this case

$$p = \lceil 2\alpha|m| \rceil \geq \lceil 2\alpha \cdot (k'_{r+1} - d_{r+1}) \rceil \geq \left\lceil 2\alpha \cdot \left(\left\lfloor \frac{d_{r+1}}{\alpha} \right\rfloor - d_{r+1} \right) \right\rceil \geq d_{r+1},$$

as desired.

□

Communication complexity. We now bound the communication complexity. We note that each message m_t in Π is sent in a padded form in the smooth protocol S^{Π} (where the padding may be empty). In what follows we bound the size of the padding. We refer to each bit of padding as a "dummy" bit.

We distinguish between the following cases.

1. The message m_t was sent all in a single round r, in Step 5, corresponding to the case that $|m_t| + d_r < k'_r$. In this case, this m_t was sent in S^{Π} via a (padded) message of the form $M_r = m_t \cdot 0^{d_r}$. Hence, the number of dummy bits sent for each such message is d_r.

2. The message m_t was sent in multiple rounds (since there was not enough budget to send it all at once), starting from round r. In this case, at first only a prefix of m_t, denoted by m'_t, is sent in Step 6, via a message of the form $M_r = m'_t \cdot 0 \cdot 1^p$, followed by the three messages $M_{r+1} = M_{r+2} = M_{r+3} = 0^p$ (which were sent in Step 2 or 3a or 3b), where $p = \min\{k'_r - 2, \lceil 2\alpha \cdot |m_t| \rceil\}$ and $|m'_t| = k'_r - p - 1$. These four messages contain $4p + 1$ dummy bits altogether.

   We distinguish between two subcases:

   (a) $\lceil 2\alpha \cdot |m_t| \rceil \leq k'_r - 2$**.** In this case,

   $$k'_{r+4} = \left\lfloor \frac{\lceil 2\alpha \cdot |m_t| \rceil}{\beta} \right\rfloor \geq \left\lfloor \frac{2\alpha \cdot |m_t|}{\beta} \right\rfloor \geq 2 \cdot |m_t|,$$

   where the last inequality follows from the fact that $\beta \leq \alpha$. In addition,

   $$d_{r+4} = \lfloor \alpha \cdot \lceil 2\alpha \cdot |m_t| \rceil \rfloor \leq \alpha \cdot |m_t|. \tag{9}$$

   Hence,

   $$|m_t| + d_{r+4} \leq |m_t| + \alpha \cdot |m_t| = (1 + \alpha) \cdot |m_t| < 2 \cdot |m_t| \leq k'_{r+4},$$

   and hence the remainder of the message m_t will be sent in round $r + 4$.

   Therefore, we conclude that in the case where $\lceil 2\alpha \cdot |m_t| \rceil \leq k'_r - 2$, the message m_t is sent with a total padding of size

   $$4p + 1 + d_{r+4} \leq 4\lceil 2\alpha \cdot |m_t| \rceil + 1 + \alpha \cdot |m_t| \leq 9\alpha \cdot |m_t| + 5.$$

We next argue that in this case,

$$\alpha \cdot |m_t| \geq 5. \tag{10}$$

This follows from the fact that

$$|m_t| \geq k'_r - d_r \geq k'_r - \alpha \cdot (k'_r + 1) \geq (1-\alpha)k'_r - \alpha$$

(where the second inequality in the equation above follows from Claim 25), and hence

$$\alpha \cdot |m_t| \geq \alpha(1-\alpha)k'_r - \alpha^2 \geq \alpha(1-\alpha) \cdot \left\lfloor \frac{1}{\beta} \right\rfloor - \alpha^2 \geq \alpha(1-\alpha) \cdot \left(\frac{1}{\beta} - 1 \right) - \alpha^2 \geq \frac{\alpha(1-\alpha)}{\beta} - 1 \geq 5,$$

as desired, where the latter follows from our assumption that $\beta \leq \frac{\alpha}{8}$ and $\alpha < \frac{1}{4}$.

Thus, in this case the total padding of m_t is bounded by

$$4p + 1 + d_{r+4} \leq 10\alpha \cdot |m_t|.$$

(b) $\lceil 2\alpha \cdot |m_t| \rceil > k'_r - 2$**.** In this case, $M_{r+1} = M_{r+2} = M_{r+3} = 0^p = 0^{k'_r - 2}$, and hence

$$k'_{r+4} = \left\lfloor \frac{k'_r - 2}{\beta} \right\rfloor$$

and

$$d_{r+4} = \lfloor \alpha \cdot (k'_r - 2) \rfloor \leq \alpha \cdot k'_r \leq \alpha \cdot \frac{\alpha \cdot |m_t|}{1-\alpha} \leq \frac{\alpha \cdot |m_t|}{3},$$

where the third equation follows from Claim 25, and the forth equation follows our assumption that $\alpha \leq \frac{1}{4}$.

In this case, denote the remainder of m_t by m''_t; namely, $m_t = (m'_t, m''_t)$.

If it holds that $m''_t + d_{r+2} < k'_{r+4}$, then as above, the remainder of the message m_t (i.e., m''_t) will be sent in round $r+4$, and the total padding for m_t will be of size

$$4p+1+d_{r+4} = 4(k'_r-2)+1+d_{r+4} \leq 4\lceil 2\alpha \cdot |m_t| \rceil + 1 + \alpha \cdot |m_t| \leq 9\alpha \cdot |m_t| + 5 \leq 10\alpha \cdot |m_t|,$$

where the latter inequality follows from Equation (10).

On the other hand, if $m''_t + d_{r+4} \geq k'_{r+4}$ then only a prefix of the remaining m_t (i.e., of m''_t) will be sent in round $r+4$, in which case we again distinguish between the two subcases, as above.

In what follows, we change notations, and denote the first prefix of m_t that was sent in round r by $m'_{t,0}$ and denote the remaining of m_t by $m''_{t,1}$ so that $m_t = (m'_{t,0}, m''_{t,1})$. More generally we denote the part of m_t that was sent in round $r+4i$ by $m'_{t,i}$, and we denote the remaining part by $m''_{t,i+1}$ so that $m''_{t,i} = (m'_{t,i}, m''_{t,i+1})$ and $m_t = (m'_{t,0}, m'_{t,1}, \ldots, m'_{t,i}, m''_{t,i+1})$. We define $m''_{t,0} = m_t$.

Let ℓ be the smallest integer for which

$$\lceil 2\alpha |m''_{t,\ell}| \rceil < k'_{r+4\ell} - 2.$$

Sending the message m_t requires either $4\ell + 1$ messages if

$$|m''_{t,\ell}| + d_{r+4\ell} < k'_{r+4\ell},$$

or $4\ell + 5$ messages if

$$|m''_{t,\ell}| + d_{r+4\ell} \geq k'_{r+4\ell},$$

where in both cases, $d_{r+4\ell} < \alpha \cdot k'_{r+4(\ell-1)}$. In both cases, the number of dummy bits in the first 4ℓ messages is

$$(4(k'_r-2)+1)+(4(k'_{r+4}-2)+1)+\ldots+(4(k'_{r+4(\ell-1)}-2)+1) \leq 4k'_r+4k'_{r+4}+\ldots+4k'_{r+4(\ell-1)}.$$

Recall that for every i for which

$$\lceil 2\alpha|m''_{t,i}|\rceil \geq k'_{r+4i} - 2$$

it holds that $|M_{r+4i+1}| = k'_{r+4i} - 2$, and hence

$$k'_{r+4(i+1)} = \left\lfloor \frac{1}{\beta} \cdot \left(k'_{r+4i} - 2\right) \right\rfloor > \frac{1}{\beta} \cdot \left(k'_{r+4i} - 2\right) - 1,$$

which implies that

$$k'_{r+4i} \leq \beta k'_{r+4(i+1)} + 3 \leq 4\beta \cdot k'_{r+4(i+1)},$$

where the latter inequality follows from the fact that $k'_{r+2(i+1)} > \frac{1}{\beta}$. Hence, we can bound the number of dummy bits in the first 4ℓ messages by

$$\begin{aligned}
&4k'_r + 4k'_{r+4} + \ldots + 4k'_{r+4(\ell-1)} < \\
&4 \cdot (k'_{r+4(\ell-1)} + k'_{r+4(\ell-2)} + \ldots + k'_r) \leq \\
&4k'_{r+4(\ell-1)} \cdot \left(1 + 4\beta + (4\beta)^2 + \ldots + (4\beta)^{\ell-1}\right) \leq \\
&4k'_{r+4(\ell-1)} \cdot \frac{1}{1-4\beta} \leq \\
&5k'_{r+4(\ell-1)},
\end{aligned}$$

where the latter inequality follows from our assumption that $\beta \leq \frac{1}{32}$ (which in turn follows from the fact that $\beta \leq \frac{\alpha}{8}$ and the fact that $\alpha \leq \frac{1}{4}$).

Recall that by the definition of ℓ, it holds that

$$\lceil 2\alpha|m_t|\rceil \geq k'_{r+4(\ell-1)} - 2 \geq \frac{3k'_{r+4(\ell-1)}}{4},$$

where the latter follows from the fact that $k'_{(r+4(\ell-1))} > \frac{1}{\beta} \geq 8$. Thus, the total number of dummy bits in the first 4ℓ rounds is at most

$$\frac{20}{3}\lceil 2\alpha|m_t|\rceil \leq 14\alpha|m_t| + 8 \leq 16\alpha|m_t|,$$

where the latter inequality follows from Inequality (10). In order to bound the number of dummy bits sent after these 2ℓ rounds of transmission of m_t, we distinguish between two cases.

Case 1: $|m''_{t,\ell}| + d_{r+4\ell} < k'_{r+4\ell}$**.** In this case, in addition to these 2ℓ messages, containing in total at most $16\alpha|m_t|$ dummy bits, there is a single additional message containing at most

$$d_{r+4\ell} \leq \lfloor \alpha \cdot k'_{r+4(\ell-1)} \rfloor \leq \lfloor \alpha|m_t| \rfloor$$

dummy bits, where the latter follows from the definition of ℓ, which implies that

$$k'_{r+4(\ell-1)} \leq \lceil 2\alpha|m_t|\rceil + 2 \leq 2\alpha|m_t| + 3 \leq 3\alpha|m_t|,$$

where the latter inequality follows from Equation (10). Thus, in this case the total number of dummy bits sent in order to send m_t is bounded by $19\alpha|m_t|$.

Case 2: $|m''_{t,\ell}| + d_{r+2\ell} \geq k'_{r+2\ell}$**.** In this case, in addition to these 4ℓ messages, containing in total at most $16\alpha|m_t|$ dummy bits, five additional messages will be sent containing in total

$$\begin{aligned}
&4p + 1 + d_{r+4(\ell+1)} \leq \\
&4\lceil 2\alpha|m_t|\rceil + 1 + d_{r+4(\ell+1)} \leq \\
&4\lceil 2\alpha|m_t|\rceil + 1 + \alpha|m_t| \leq \\
&9|\alpha|m_t| + 3 \leq \\
&10|\alpha|m_t|
\end{aligned}$$

dummy bits, where the second inequality follows from Equation (9) and the last inequality follows from Equation (10). Thus, in this case the total number of dummy bits sent in order to send m_t is bounded by $26\alpha|m_t|$.

We conclude that each message m_t that requires multiple rounds to be sent contributes to at most $26\alpha|m_t|$ dummy bits. Moreover, recall that each message that is sent in a single round r contributes at most d_r dummy bits, and thus, all these dummy bits together can blowup the communication complexity by at most 3α. Therefore we conclude that the total communication of S^{Π} is bounded by

$$\mathrm{CC}(S^{\Pi}) \leq \mathrm{CC}(\Pi) \cdot (1 + 26\alpha) \cdot (1 + 3\alpha) = 1 + 29\alpha + 26 \cdot 3 \cdot \alpha^2 \leq 1 + 29\alpha + \alpha \cdot \frac{3 \cdot 26}{4} \leq 1 + 50\alpha,$$

as desired, where the second to last inequality follows from our assumption that $\alpha \leq \frac{1}{4}$.

Round complexity Finally, we are ready to bound the round complexity of S^{Π}. Let

$$r_1, r_2, ..., r_k$$

be a partition of the protocol where r_i is the first round where the message m_i was sent, and where k denotes the round complexity of Π. Thus, the message m_i was delivered in $r_{i+1} - r_i$ rounds. As we saw, the messages which deliver m_i consist of the following messages:

- **Case 1:** $m_i + d_i < k_i$**.** In this case m_i is delivered via a single message, and hence $r_{i+1} = r_i + 1$.
- **Case 2:** $m_i + d_i \geq k_i$**.** In this case m_i is delivered via $4(\ell+1)+1$ messages, where $\ell \geq 0$, where the first $4(\ell+1)$ messages are of size

  $$k'_{r_i},\ (k'_{r_i} - 2)^{\times 3},\ k'_{r_i+4},\ (k'_{r_i+4} - 2)^{\times 3},\ \ldots,\ k'_{r_i+4(\ell-1)},\ (k'_{r_i+4(\ell-1)} - 2)^{\times 3},\ k'_{r_i+4\ell},\ p^{\times 3},$$

  where $p \leq k'_{r_i+4\ell} - 2$, followed by a single message of length $< k'_{r_i+4(\ell+1)}$, and where in the above $w^{\times 3} \triangleq (w, w, w)$.

Note that for every $j \in [\ell]$,

$$k'_{r_i+4j} = \left\lfloor \frac{k'_{r_i+4(j-1)} - 2}{\beta} \right\rfloor \geq \frac{k'_{r_i+4(j-1)}}{2\beta},$$

and

$$k'_{r_i} = \left\lfloor \frac{1}{\beta} \cdot \min\left\{|M_{r_i-1}|, |M_{r_i-2}|, |M_{r_i-3}|\right\} \right\rfloor \geq \left\lfloor \frac{1}{\beta} \cdot \beta \cdot |M_{r_i-1}| \right\rfloor \geq |M_{r_i-1}|,$$

where the latter follows from β-smoothness of S^{Π}.

Therefore, the number of messages between rounds r_{i+1} and r_i is bounded by

$$1+4\cdot\log_{\frac{1}{2\beta}}\left(\frac{k'_{r_i+4\ell}}{k'_{r_i}}\right)\leq 1+4\cdot\log_{\frac{1}{2\beta}}\left(\frac{k'_{r_i+4\ell}}{|M_{r_i-1}|}\right)\leq 1+4\cdot\log_{\frac{1}{2\beta}}\left(\frac{|M_{r_{i+1}-1}|}{\alpha^2\cdot|M_{r_i-1}|}\right),$$

where the latter inequality follows from α-smoothness.

Combining the above observations, we get the following upper bound on the total number of rounds,

$$\begin{aligned}
&R(S^{\Pi})\leq\\
&\sum_{j=1}^{k}\left(1+4\log_{\frac{1}{2\beta}}\left(\frac{|M_{r_{i+1}-1}|}{\alpha^2\cdot|M_{r_i-1}|}\right)\right)=\\
&k+\sum_{j=1}^{k}4\log_{\frac{1}{2\beta}}\left(\frac{|M_{r_{i+1}-1}|}{\alpha^2\cdot|M_{r_i-1}|}\right)\leq\\
&k+\sum_{j=1}^{k}4\log_{\frac{1}{2\beta}}\frac{|M_{r_{i+1}-1}|}{|M_{r_i-1}|}+\sum_{j=1}^{k}4\log_{\frac{1}{2\beta}}\alpha^{-2}\leq\\
&k+\sum_{j=1}^{k}4\log_{\frac{1}{2\beta}}\frac{|M_{r_{i+1}-1}|}{|M_{r_i-1}|}+8k\log_{2\beta}\alpha\leq\\
&k+4\log_{\frac{1}{2\beta}}\prod_{i=1}^{k}\frac{|M_{r_{i+1}-1}|}{|M_{r_i-1}|}+8k\log_{2\beta}\alpha\leq\\
&k+4\log_{\frac{1}{2\beta}}\mathrm{CC}(S^{\Pi})+8k\log_{2\beta}\alpha\leq\\
&k\cdot(1+8\log_{2\beta}\alpha)+4\log_{\frac{1}{2\beta}}\cdot\mathrm{CC}(\Pi)\cdot(1+50\alpha)\leq\\
&R(\Pi)\cdot(1+8\log_{2\beta}\alpha)+4\log_{\frac{1}{2\beta}}\cdot\mathrm{CC}(\Pi)+4\log_{\frac{1}{2\beta}}(1+50\alpha)\leq\\
&R(\Pi)\cdot(1+8\log_{2\beta}\alpha)+4\log_{\frac{1}{2\beta}}\cdot\mathrm{CC}(\Pi)+4,
\end{aligned}$$

as desired.

□

B Proof of Theorem 9.

In the proof of Theorem 9 we use the following terminology. We define the terminology from Alice's point of view. We use analogous terminology for Bob. For each round r, we denote by $m_{A,r}^{\mathsf{sent}}$ the message that Alice sent in round r, and we denote by $m_{A,r}^{\mathsf{receive}}$ the message that Bob thinks Alice sent in round r.

In the analysis, we denote

$$|m_{A,r}|=\max\left\{\left|m_{A,r}^{\mathsf{sent}}\right|,\left|m_{A,r}^{\mathsf{receive}}\right|\right\},$$

and we denote

$$\ell_{A,r}^{\mathsf{sent}}=|m_{A,r}^{\mathsf{sent}}|\ ,\ \ \ell_{A,r}^{\mathsf{receive}}=|m_{A,r}^{\mathsf{receive}}|\ ,\ \text{ and }\ \ \ell_{A,r}=\max\{\ell_{A,r}^{\mathsf{sent}},\ell_{A,r}^{\mathsf{receive}}\}.$$

We use similar notations for Bob, and we let

$$\ell_{\max,r} = \max\{\ell_{A,r}, \ell_{B,r}\},$$

so $\ell_{\max,r}$ is the maximal size of message sent or received in round r. We emphasize that these notations were used with a different meaning in the protocol, where error was not considered. In the protocol, $\ell_{A,r}$ was the length of the message sent by Alice in round r, $\ell_{B,r}$ was the length of the message that Alice received from Bob in round r (recall that the protocol was defined from the perspective of Alice), and $\ell_{\max,r} = \max\{\ell_{A,r}, \ell_{B,r}\}$. Using the new notation, the values ℓ_r^- and ℓ_r^+ may be different for Alice and Bob. In particular, using our new notation:

$$\ell_{A,r}^+ = \min\left\{\frac{\ell_{A,r}^{\mathsf{sent}}}{\beta}, 2\max\left\{\ell_{A,r}^{\mathsf{sent}}, \ell_{B,r}^{\mathsf{receive}}\right\}\right\}$$

and

$$\ell_A^- = \min\left\{\frac{\ell_{A,r}^{\mathsf{sent}}}{\beta}, \max\left\{\beta^{-1}, \alpha\max\left\{\ell_{A,r}^{\mathsf{sent}}, \ell_{B,r}^{\mathsf{receive}}\right\}\right\}\right\}.$$

The values for Bob are defined analogously.

We say that a message is corrupted if the adversary corrupts the message or if there is a hash collision in the hash associated with the message. We denote by E the set of all messages $m_{A,r}$ or $m_{B,r}$ that were corrupted (due to adversarial error or due to hash collisions).

Given any ordered set of messages T, where $T = (m_{A,r}, m_{B,r})_{r=1}^t$ for some $t \in \mathbb{N}$, we denote by

$$|T| = \sum_{m_{A,r} \in T} |m_{A,r}| + \sum_{m_{B,r} \in T} |m_{B,r}|$$

and refer to $|T|$ as the volume of T. We denote the number of messages in T by $|T|'$. Recall that T corresponds to a prefix of the transcript, and thus T' is twice the number of rounds in T (since two messages are sent in each round, one by Alice and one by Bob).

Partitioning the protocol into chunks. We partition the error-resilient protocol into chunks, as follows.

1. **Good** Simulation **chunks:** These chunks consist of all consecutive rounds where both parties are in Simulation State and indeed $T_A = T_B$.
2. **Good** Verification **chunks:** These chunks consist of all consecutive rounds where both parties are in Verification State and indeed $T_A[R_A] = T_B[R_B]$.
3. **Good** Correction **chunks:** These chunks begin when $S_A = S_B = r$ for some r, and end when at least one party changes the value of its state S.
4. **Bad** Correction **chunks:** These chunks consist of all consecutive rounds where $S_A \neq S_B$, and at least one of the parties is in Correction State (i.e. $S \in \mathbb{N}$).
5. **Bad chunks:** These chunks consist of all consecutive rounds such that $S_A \neq S_B$ or $T_A[R_A] \neq T_B[R_B]$, and none of the parties are in Correction State.

The bulk of the technical difficulty is in proving the following lemma.

Lemma 29. *The number of rounds in the (error-resilient) protocol* (S^A, S^B) *that are not in a Good* Simulation *chunk is at most*

$$301d \log \beta^{-1} |E|',$$

and the volume of all the messages that are not in a Good Simulation *chunk is at most*

$$7\beta^{-1}|E| + 10d\beta^{-1}|E|'.$$

The remaining of the Section is devoted to the proof of Lemma 29. We start with the following claim (which will be used later in this section).

Claim 30. *For any round* $r > 1$*, the protocol satisfies*

$$|m^{\mathsf{sent}}_{A,r}| \leq \beta^{-1} \min\{|m^{\mathsf{sent}}_{A,r-1}|, |m^{\mathsf{receive}}_{B,r-1}|\}$$

and

$$|m^{\mathsf{sent}}_{B,r}| \leq \beta^{-1} \min\{|m^{\mathsf{sent}}_{B,r-1}|, |m^{\mathsf{receive}}_{A,r-1}|\}$$

Moreover,

$$|m^{\mathsf{sent}}_{A,r}| \geq \alpha \max\{|m^{\mathsf{sent}}_{A,r-1}|, |m^{\mathsf{receive}}_{B,r-1}|\}$$

unless $S_{A,r-1} =$ Verification *and* $S_{A,r} =$ Simulation*, and similarly*

$$|m^{\mathsf{sent}}_{B,r}| \geq \alpha \max\{|m^{\mathsf{sent}}_{B,r-1}|, |m^{\mathsf{receive}}_{A,r-1}|\}$$

unless $S_{B,r-1} =$ Verification *and* $S_{B,r} =$ Simulation*.*

Proof. In what follows, we bound $|m^{\mathsf{sent}}_{A,r}|$. The bound on $|m^{\mathsf{sent}}_{B,r}|$ is obtained in a similar manner. If $S_{A,r-1} \in \mathbb{N}$ (i.e., Alice was in Correction state), then

$$|m^{\mathsf{sent}}_{A,r}| = \ell^-_{r-1},$$

and by definition of ℓ^-_{r-1},

$$\alpha \max\{|m^{\mathsf{sent}}_{A,r-1}|, |m^{\mathsf{receive}}_{B,r-1}|\} \leq \ell^-_{r-1} \leq \beta^{-1}|m^{\mathsf{sent}}_{A,r-1}|$$

as desired.

If $S_{A,r-1} =$ Verification and $S_{A,r} \in \{\text{Verification}, r\}$, then in round r she sent a message of length at most

$$\ell^+_{r-1} = \min\left\{\beta^{-1}\ell^{\mathsf{sent}}_{A,r-1}, 2\ell^{\mathsf{receive}}_{B,r-1}\right\} \leq \beta^{-1} \min\left\{\ell^{\mathsf{sent}}_{A,r-1}, \ell^{\mathsf{receive}}_{B,r-1}\right\},$$

and a message of length at least

$$\ell^-_{r-1} \leq \alpha \max\{|m^{\mathsf{sent}}_{A,r-1}|, |m^{\mathsf{receive}}_{B,r-1}|\},$$

where $\ell^+_{r-1} \geq \ell^-_{r-1}$, as desired. If $S_{A,r-1} =$ Simulation then by the smoothness of the underlying protocol,

$$|m^{\mathsf{sent}}_{A,r}| \leq \max\left\{\ell^-_{r-1}, \beta^{-1} \min\{|m^{\mathsf{sent}}_{A,r-1}|, |m^{\mathsf{receive}}_{B,r-1}|\}\right\} \leq \beta^{-1} \min\{|m^{\mathsf{sent}}_{A,r-1}|, |m^{\mathsf{receive}}_{B,r-1}|\},$$

and

$$|m^{\mathsf{sent}}_{A,r}| \geq \min\left\{\ell^-_{r-1}, \alpha \max\{|m^{\mathsf{sent}}_{A,r-1}|, |m^{\mathsf{receive}}_{B,r-1}|\}\right\} \geq \alpha \max\{|m^{\mathsf{sent}}_{A,r-1}|, |m^{\mathsf{receive}}_{B,r-1}|\},$$

as desired.

If $S_{A,r-1} =$ Verification and $S_{A,r} =$ Simulation, then $\left|m^{\mathsf{sent}}_{A,r}\right| \leq \left|m^{\mathsf{sent}}_{A,r-1}\right|$, since by the definition of the protocol, in this case $|m^{\mathsf{sent}}_{A,r-1}| \geq |T \backslash T[R]|$ and $m^{\mathsf{sent}}_{A,r}$ is the first message in $T \setminus T[R]$.

□

Claim 31. *Let C be a set of messages sent in consecutive rounds. We abuse notation and also denote by C the set of rounds in which these messages were sent. let r_0 be the first round of C, and let E_0 be the set of corrupted messages in $C \cup \{r_0 - 1\}$. Assume that for each round $r \in C$, it hold that $S_{A,r} \neq$ Simulation and $S_{B,r} \neq$ Simulation, and that each round $r \in C \setminus \{r_0\}$ has the property that if Alice (resp., Bob) did not receive a corrupted message in round $r-1$ then it sends a message of length $\ell^-_{A,r-1}$ (resp., $\ell^-_{B,r-1}$) in round r. Then*

$$|C| \leq \beta^{-1}|C|' + \left(3\beta^{-1} + 3\right)|E_0 \cap C| + 3\ell^{\mathsf{sent}}_{\max,r_0}.$$

Proof. We partition C into sub-chunks $C_1, ..., C_k$ such that the following two conditions are satisfied.

1. For every $i \in \{1, \ldots k\}$ and every round $r \in C_i$ which is not the first round in C_i, it holds that $\ell_{\max,r} = \ell^-_{A,r-1} = \ell^-_{B,r-1}$, where recall that

$$\ell^-_{A,r-1} = \min\left\{\beta^{-1}\ell^{\mathsf{sent}}_{A,r-1}, \max\{\beta^{-1}, \alpha \max\{\ell^{\mathsf{sent}}_{A,r-1}, \ell^{\mathsf{receive}}_{B,r-1}\}\right\}.$$

2. For every $i \in \{2, \ldots, k\}$, the first round $r' \in C_i$ satisfies $\ell_{\max,r'} \neq \ell^-_{A,r'-1}$ or $\ell_{\max,r'} \neq \ell^-_{B,r'-1}$.

We will bound

$$|C_i| \leq \beta^{-1}|C_i|' + 3|E_0 \cap C_i| + 3\beta^{-1}|E_0 \cap C_{i-1}|, \tag{11}$$

and

$$|C_1| \leq \beta^{-1}|C_1|' + 3|E_0 \cap C_1| + 3\ell^{\mathsf{sent}}_{\max,r_0}. \tag{12}$$

This is sufficient since:

$$\begin{aligned}
|C| &= \sum_{i=1}^{k} |C_i| \\
&\leq \sum_{i=1}^{k} \beta^{-1}|C_i|' + \sum_{i=2}^{k} 3\beta^{-1}|E_0 \cap C_{i-1}| + \sum_{i=1}^{k} 3|E_0 \cap C_i| + 3\ell^{\mathsf{sent}}_{\max,r_0} \\
&\leq \beta^{-1}|C|' + (3\beta^{-1} + 3) \cdot |E_0| + 3\ell^{\mathsf{sent}}_{\max,r_0}.
\end{aligned}$$

It thus remains to prove Equations (11) and (12). To this end, fix any $i \in \{1, \ldots, k\}$. Let r' be the first round in the sub-chunk C_i. We will show by induction that each round $r \in C_i$ satisfies $\ell_{\max,r} \leq \max\left\{\beta^{-1}, \alpha^{r-r'}\ell_{\max,r'}\right\}$. Indeed, the statement is true for $r = r'$. For $r > r'$, by the induction hypothesis, we get,

$$\begin{aligned}
\ell_{\max,r} &= \ell^-_{A,r-1} \leq \max\{\beta^{-1}, \alpha\ell_{\max,r-1}\} \\
&\leq \max\left\{\beta^{-1}, \alpha \cdot \alpha^{r-1-r'}\ell_{\max,r'}\right\} \\
&= \max\left\{\beta^{-1}, \alpha^{r-r'}\ell_{\max,r'}\right\}.
\end{aligned}$$

So we can bound $|C_i|$ using $\ell_{\max,r'}$ as follows.

$$\begin{aligned}
|C_i| &\leq \sum_{r\in C_i} 2\ell_{\max,r} \leq \sum_{r\in C_i} 2\max\left\{\beta^{-1}, \alpha^{r-r'}\ell_{\max,r'}\right\} \\
&\leq \sum_{r\in C_i}\left(2\beta^{-1} + 2\alpha^{r-r'}\ell_{\max,r'}\right) \\
&\leq \beta^{-1}|C_i|' + 2\ell_{\max,r'}\sum_{r\in C_i}\alpha^{r-r'} \leq \\
&\leq \beta^{-1}|C_i|' + \frac{2\ell_{\max,r'}}{1-\alpha} \leq \\
&\leq \beta^{-1}|C_i|' + 3\ell_{\max,r'}
\end{aligned}$$

To prove Equations (11) and (12) it suffices to bound $\ell_{\max,r'}$. First assume that $\ell_{\max,r'} \neq \ell^{\mathsf{sent}}_{\max,r'}$. In this case, $\ell_{\max,r'} \leq |E_0 \cap C_i|$. This, together with the above calculations, implies that $|C_i| \leq \beta^{-1}|C_i|' + 3|C_i \cap E_0|$, which in turn implies that Equations (11) and (12) hold.

From now on assume that $\ell_{\max,r'} = \ell^{\mathsf{sent}}_{\max,r'}$. In this case Equation (12) holds. To prove Equation (11), we prove that for every $C_i \neq C_1$,

$$\ell_{\max,r'} \leq \beta^{-1}|E_0 \cap C_{i-1}| \tag{13}$$

To this end, note that if one of the message sent in round $r'-1$ was corrupted, then Equation (13) follows by the smoothness property (see Claim 30). Thus, from now on, we assume that the messages sent in round $r'-1$ were not corrupted, which in turn implies that in round r' the parties send messages of length $\ell^-_{A,r'-1}$ and $\ell^-_{B,r'-1}$. Moreover, by the definition of r', it must be the case that $\ell^-_{A,r'-1} \neq \ell^-_{B,r'-1}$.

The fact that the messages sent in round $r'-1$ were not corrupted implies that

$$\max\{\ell^{\mathsf{sent}}_{A,r'-1}, \ell^{\mathsf{receive}}_{B,r'-1}\} = \max\{\ell^{\mathsf{sent}}_{B,r'-1}, \ell^{\mathsf{receive}}_{A,r'-1}\}.$$

This, together with the assumption that $\ell^-_{A,r'-1} \neq \ell^-_{B,r'-1}$, implies that

$$\ell_{\max,r'} > \ell^-_{A,r'-1} = \beta^{-1}\ell^{\mathsf{sent}}_{A,r'-1} < \alpha\max\{\ell^{\mathsf{sent}}_{A,r'-1}, \ell^{\mathsf{receive}}_{B,r'-1}\},$$

or

$$\ell_{\max,r'} > \ell^-_{B,r'-1} = \beta^{-1}\ell^{\mathsf{sent}}_{B,r'-1} < \alpha\max\{\ell^{\mathsf{sent}}_{B,r'-1}, \ell^{\mathsf{receive}}_{A,r'-1}\}.$$

Suppose without loss of generality that the former holds. This implies that

$$\ell^{\mathsf{sent}}_{A,r'-1} < \alpha\beta\cdot\ell^{\mathsf{receive}}_{B,r'-1} = \alpha\beta\cdot\ell^{\mathsf{sent}}_{B,r'-1} \leq \alpha\ell^{\mathsf{sent}}_{B,r'-2},$$

where the last inequality follows from Claim 30. By the definition of C, $S_{A,r'-1} \neq$ Simulation. This, together with Claim 30, implies $\ell^{\mathsf{sent}}_{A,r'-1} \geq \alpha\cdot\ell^{\mathsf{receive}}_{B,r'-2}$. Thus, the message sent by Bob in round $r'-2$ must have been corrupted. Therefore,

$$\begin{aligned}
&\ell_{\max,r'} = \\
&\max\{\ell^-_{A,r'-1}, \ell^-_{B,r'-1}\} \leq \\
&\max\{\beta^{-1}, \alpha\max\{\ell^{\mathsf{sent}}_{A,r'-1}, \ell^{\mathsf{sent}}_{B,r'-1}\}\} \leq \\
&\max\{\beta^{-1}, \alpha\beta^{-1}\cdot\ell_{B,r'-2}\} \leq \\
&\beta^{-1}|E_0 \cap C_{i-1}|,
\end{aligned}$$

as desired, where the first equation follows from our assumption that the messages in round $r'-1$ were not corrupted (together with the definition of C); the second equation follows from the definition of ℓ^-; the third equation follows from the smoothness condition (Claim 30); and the last equation follows from the fact that the message that Bob sent in round $r'-2$ was corrupted.

□

In the following claims, we bound the volume and the number of messages in each chunk, as a function of $|E|$ and $|E|'$ (where recall that E is the set of all corrupted messages, m_A and m_B).

Lemma 32. *The total volume of all the Bad chunks is bounded by $(2\beta^{-1}+1)|E|+3d\beta^{-1}|E|'$, and the total number of messages exchanged in all the Bad chunks is bounded by $5d|E|'$.*

Proof. Fix a bad chunk C and let E_0 be the set of corrupted messages in this chunk and in the chunk preceding it. Note that by definition, the chunk preceding a bad chunk is never a bad chunk. Hence, it suffices to prove that

$$|C| \leq (2\beta^{-1}+1)|E_0|+3d\beta^{-1}|E_0|'$$

and

$$|C|' \leq 5d|E_0|'.$$

We first prove that $|C|' \leq 5d|E_0|'$. To this end, we partition the messages in C into two sets, those that are sent with a hash, and those that are sent without a hash, and we denote the former by $C \cap H$.

Note that there is at most one round in the chunk C that has a non corrupted message m sent with a hash, since after such a round at least one party will change its state into Correction State. Therefore, $|(C \cap H) \setminus E_0|' \leq 2$. Hence,

$$|C \cap H|' = |(C \cap H) \cap E_0|' + |(C \cap H) \setminus E_0|' \leq |E_0|' + 2 \leq 3|E_0|',$$

where the latter inequality follows from the fact that $|E_0|' \geq 1$, which in turn follows from the fact that if the previous chunk was a Good Simulation chunk, a Good Verification chunk, a Good Correction chunk, or a Bad Correction chunk, and if there were no errors during that chunk (i.e., the messages m_A and m_B were not corrupted and there was no collisions in the corresponding hashes), then the next chunk would not be a Bad chunk. Note that, by definition the previous chunk cannot be a bad chunk (otherwise, it would be part of the Bad chunk C).

By the definition of H, for every round r for which r is multiple of d, the messages in round r are in H. Hence,

$$|C|' \leq d|C \cap H|' + 2d \leq 3d|E_0|' + 2d \leq 5d|E_0|',$$

as desired.

We next prove that

$$|C| \leq (2\beta^{-1}+1)|E_0|+3d\beta^{-1}|E_0|'.$$

To this end, we partition C into three parts: The corrupted messages in C, the non-corrupted messages in C without a hash, and the non-corrupted messages in C sent with a hash. Namely,

$$|C| = |C \cap E_0| + |C \setminus (E_0 \cup H)| + |(C \cap H) \setminus E_0|.$$

Note that

$$|C \cap E_0| \leq |E_0|$$

and

$$|C \setminus (E_0 \cup H)| \leq d \cdot |C|' \leq 5d^2|E_0|' \leq d\beta^{-1}|E_0|',$$

where this equation follows from the fact that each (uncorrupted) message without a hash is of size at most d, together with the fact that $d \leq \frac{1}{5\beta}$ (see Equation (3)).

We next bound $|(C \cap H) \setminus E_0|$. As mentioned above, $|(C \cap H) \setminus E_0|' \leq 2$. Thus, it suffices to prove that each message m in $(C \cap H) \setminus E_0$ is of size at most

$$|m| \leq \max\left\{\beta^{-1}|E_0|, d\beta^{-1}\right\}.$$

Fix any message $m \in (C \cap H) \setminus E_0$. Recall that the message m is sent in the last round of C (since as we mentioned previously, after an uncorrupted message is sent with a hash the chunk must end). We denote this last round by r_f.

Suppose that m was sent by Alice (the case that m was sent by Bob is analogous). We use the smoothness of the error-resilient protocol (Claim 30) to bound $|m|$, as follows: If $|m| \geq d\beta^{-1}$ then by smoothness, $|m^{\mathsf{receive}}_{B,r_f-1}| \geq \beta|m| \geq d$ (where recall that $m^{\mathsf{receive}}_{B,r_f-1}$ denotes the message that Alice received before sending m).

We distinguish between three cases.

1. $r_f - 1 \in C$**.** In this case, the message m_{B,r_f-1} must have been corrupted (i.e., in E_0), since otherwise, in round r_f Alice would change her state to Correction. Therefore,

   $$|m| \leq \beta^{-1}|m^{\mathsf{receive}}_{B,r_f-1}| \leq \beta^{-1} \cdot |E_0|,$$

   as desired.

2. **Round $r_f - 1$ belongs to a Good** Simulation **or Good** Verification **chunk.** In this case, the message m_{A,r_f-1} or m_{B,r_f-1} must have been corrupted, since otherwise, round r_f would have also belonged to either a Good Simulation or Verification chunk (round r_f will either remain in the same chunk as $r_f - 1$ or would be in a Simulation chunk while $r_f - 1$ is in a Verification chunk). Hence, as before,

   $$|m| \leq \beta^{-1} \min\{|m^{\mathsf{sent}}_{A,r_f-1}|, |m^{\mathsf{receive}}_{B,r_f-1}|\} \leq \beta^{-1} \cdot |E_0|,$$

   as desired.

3. **Round $r_f - 1$ belongs to a Bad** Correction **or Good** Correction **chunk.** Recall that in both a Bad Correction chunk and a Good Correction chunk, at least one of the parties is in a Correction State. Assume w.l.og. that this party is Alice, and denote $r_0 = S_{A,r_f-1}$ (i.e., the round where Alice entered this Correction State). In the next round, neither parties are in Correction state since the next round belongs to the Bad chunk C. Therefore, Alice must have changed her state into a Verification state.

   We will show that in one of the rounds between (and including) r_0 and r_f there was a corrupted message. Indeed, if the chunk preceding C was a Good Correction chunk and none of these messages were corrupted, then both parties will switch into the Verification State with matched transcripts, i.e. into a Good Verification chunk. On the other hand, if the chunk preceding C was a Bad Correction chunk, and none of these messages were corrupted, then Alice will remain in the Correction State (and the next chunk will be a Good Correction chunk). Either way, the next chunk would not be a Bad chunk. Thus, we conclude that one of the messages sent between (and including) round r_0 and r_f must have been corrupted.

Let $r' \in [r_0, r_f - 1]$ be the last round that had a corrupted message. We will show that for every $r \in [r' + 1, r_f]$ both messages sent in round r have length $\leq \beta^{-1}|E_0|$. In particular this will imply that $|m| \leq \beta^{-1} \cdot |E_0|$, as required.

The proof is by induction on the round r.

Base Case: $r = r' + 1$**.** By smoothness, we have that the size of all messages in the round $r' + 1$ is bounded by $\beta^{-1} \min\{|m_{A,r'}|, |m_{B,r'}|\} \leq \beta^{-1}|E_0|$, and hence the base case follows.

Induction Step: Consider a round $r \in [r' + 2, r_f]$. Since Alice is in a Correction State (or in the first round of a Verification state, if $r = r_f$), we have that $\ell_{A,r} = \ell^-_{A,r-1}$. We will next show that $\ell_{B,r} = \ell^-_{B,r-1}$. If in round $r - 1$ Bob was in a Simulation or a Verification State, then since he received an uncorrupted message from Alice (which includes the fact that she is in a Correction State), he would change his state into a Correction State, and send a message of length $\ell^-_{B,r-1}$. On the other hand, if he was in a Correction State, he would also send a message of length $\ell^-_{B,r-1}$ (by definition). This, together with the fact that the messages in round $r - 1$ and round r were not corrupted, and with our induction hypothesis, implies that

$$\ell_{A,r}, = \ell^-_{A,r-1} = \min\{\beta^{-1}\ell_{A,r-1}, \max\{\beta^{-1}, \alpha\ell_{\max,r-1}\} \leq \max\{\beta^{-1}|E_0|, \ell_{\max,r-1}\} \leq \beta^{-1}|E_0| \ ,$$

and

$$\ell_{B,r}, = \ell^-_{B,r-1} = \min\{\beta^{-1}\ell_{B,r-1}, \max\{\beta^{-1}, \alpha\ell_{\max,r-1}\} \leq \max\{\beta^{-1}|E_0|, \ell_{\max,r-1}\} \leq \beta^{-1}|E_0| \ ,$$

as desired.

□

Lemma 33. *The total number of messages exchanged in all Bad* Correction *chunks is bounded by* $20|E|'$.

Proof. Fix a Bad Correction chunk C. Denote its first round by r_0 and denote its last round by r_f. Let E_0 be the set of corrupted messages in all the chunks starting after the previous Bad Correction chunk until (and including) chunk C. Note that it is enough to show that $|C|' \leq 20|E_0|'$.

First we argue that $|E_0|' \geq 1$. To this end, assume that $|E_0|' = 0$, and we will get a contradiction by showing that the protocol cannot reach a Bad Correction chunk if there were no errors since the end of the previous Bad Correction chunk. This follows from the following simple claims.

1. Between the previous Bad Correction chunk and C, the protocol cannot be in a Good Simulation chunk, since without errors, the protocol will remain in a Good Simulation chunk.

2. Between the previous Bad Correction chunk and C, the protocol cannot be in a Good Verification chunk, since without errors, the protocol will move into a Good Simulation chunk.

3. Between the previous Bad Correction chunk and C, the protocol cannot be in a Good Correction chunk, since without errors, it will move into a Good Verification chunk.

4. Between the previous Bad Correction chunk and C, the protocol cannot be in a Bad chunk, since if the protocol did enter a Bad chunk after the previous Bad Correction chunk it must be the case that one party changed its state from a Correction state into a Verification state. Without corruptions, after one round in the Bad chunk, this party gets either a message of length $< \beta^{-1}$, or a message of length $\geq \beta^{-1} > d$ with a hash. Either way, she will detect this inconsistency and change her state into a Correction state. The other party will receive a message of length $\geq \beta^{-1}$ with a hash, and thus will also change its state into a Correction state. Since both parties change their state into a Correction state at the same round, the protocol will enter a Good Correction chunk, after which the parties will not move into a Bad Correction chunk, unless errors occurred.

5. It cannot be the case that there is no chunk between the previous Bad Correction chunk and chunk C, since in this case the previous Bad Correction chunk and chunk C would have been combined into a single Bad Correction chunk.

We thus conclude that $|E_0|' \geq 1$.

We next prove that $|C|' \leq 20|E_0|'$. The fact that $|E_0|' \geq 1$ implies that it suffices to prove that $|C|' \leq 18|E_0|' + 2$. We prove the latter using a potential argument. We define for every $r \in C$,

$$\Psi_A(r) = \begin{cases} r - S_{A,r} - \frac{4}{3}v^0_{A,r} & S_{A,r} \in \mathbb{N} \\ 0 & S_{A,r} \in \{\text{Simulation}, \text{Verification}\} \end{cases},$$

where $v^0_{A,r}$ is the value of the variable v^0 of Alice at the end of round r. We define Ψ_B similarly.

Our potential function Φ is defined as follows: For any $r \in C$,

$$\Phi(r) = |E_0(r)|' - \frac{1}{3}\Psi_A(r) - \frac{1}{3}\Psi_B(r) ,$$

where $E_0(r)$ consists with all messages in E_0 before (not including) round r.

In order to bound $|C|'$, we will show three properties of Φ:

1. Upper-bound: $\Phi(r) \leq |E_0(r)|'$ for every $r \in C$.
2. Non-negative: $\Phi(r_0) \geq 0$.
3. Increasing: $\Phi(r) - \Phi(r-1) \geq \frac{1}{9}$ for every $r \in C \setminus \{r_0\}$.

Indeed this will bound $|C|' \leq 18|E_0|' + 2$ since,

$$|E_0|' \geq |E_0(r_f)|' \geq \Phi(r_f) = (\Phi(r_f) - \Phi(r_0)) + \Phi(r_0) \geq \frac{1}{9}(r_f - r_0) + 0 = \frac{1}{9}\left(\frac{1}{2}|C|' - 1\right) ,$$

where the latter equality follows from the fact that the number of rounds in C, which is equal to $\frac{1}{2}|C|'$, is $r_f - r_0 + 1$.

Upper-bound: $\Phi(r) \leq |E_0(r)|'$. We prove this by showing that for any r,

$$\Psi_A(r) \geq 0, \tag{14}$$

and similarly for Bob. By definition, when $S_{A,r} \in \{\text{Simulation}, \text{Verification}\}$ then $\Psi_A(r) = 0$, and the claim follows.

Next, consider the case where $S_{A,r} = r' \in \mathbb{N}$, and we will show that

$$v^0_{A,r} \leq \frac{3}{4}(r - r').$$

This suffices since $\Psi_A = r - r' - \frac{4}{3}v^0_{A,r} \geq 0$, as desired.

Let q be a power of two that satisfies $\frac{1}{2}(r - r') \leq q < (r - r')$. We distinguish between two cases:

Case 1: $v^0_{A,r'+q-1} > \frac{1}{2}q$. In this case, by the definition of the protocol Π, in Step 3c of the protocol, $v^0_{A,r'+q}$ would have been set to 0, and hence

$$v^0_{A,r} \leq \; v^0_{A,r'+q} + (r - r' - q) \leq 0 + (r - r' - q) \leq \frac{1}{2}\left(r - r'\right) \leq \frac{3}{4}\left(r - r'\right) \;,$$

as desired, where the first inequality follows from the fact that in each round $v^0_{A,r}$ increases by at most 1.

Case 2: $v^0_{A,r'+q-1} \leq \frac{1}{2}q$. In this case,

$$v^0_{A,r} \leq \; v^0_{A,r'+q-1} + (r - r' - q + 1) \leq \frac{1}{2}q + (r - r' - q + 1) = (r - r' + 1) - \frac{1}{2}q \leq \frac{3}{4}\left(r - r'\right) \;,$$

as desired, where the first inequality follows from the fact that in each round $v^0_{A,r}$ increases by at most 1.

Non-negativity of Φ: Here we show that $\Phi(r_0) \geq 0$.

We first note that for every round r,

$$S_{A,r} \in \{\text{Simulation}, \text{Verification}, r\} \Rightarrow \Psi_A(r) = 0. \tag{15}$$

Indeed, if $S_{A,r} \in \{\text{Simulation}, \text{Verification}\}$ then $\Psi_A(r) = 0$ by definition. If $S_{A,r} = r$ then $v^0_{A,r} = 0$ (see Step 3c in the protocol), and hence $\Psi_A(r) = r - r - 0 = 0$.

To prove that $\Phi(r_0) \geq 0$, we first consider the case where the previous chunk was not a Good Correction chunk (i.e., in the previous chunk the state of both parties is Simulation or Verification). In this case, both parties satisfy $S_{A,r_0}, S_{B,r_0} \in \{\text{Simulation}, \text{Verification}, r_0\}$. This, together with Equation (15), implies that $\Psi_A(r_0) = \Psi_B(r_0) = 0$, and hence $\Phi(r_0) = |E_0(r_0)| - \frac{1}{3}\Psi_A(r_0) - \frac{1}{3}\Psi_B(r_0) \geq 0$.

Next, consider the case where the previous chunk was a Good Correction chunk. Namely, for some r' we get that $S_{A,r_0-1} = S_{B,r_0-1} = r'$. Note that in round r_0, the parties states are in disagreement (since it is a Bad Correction chunk), hence it must be the case that $|E_0(r_0)|' \geq \frac{1}{2}(r_0 - r')$.

Note that in round r_0 one of the parties (say Alice) changes her state to $S_{A,r_0} \in \{r_0, \text{Verification}\}$. This, together with Equation (15), implies that $\Psi_A(r_0) = 0$. Moreover, $\Psi_B(r_0) \leq (r_0 - r')$. Hence,

$$\Phi(r_0) = |E_0(r_0)|' - \frac{1}{3}\Psi_A(r_0) - \frac{1}{3}\Psi_B(r_0) \geq \frac{1}{2}(r_0 - r') - \frac{1}{3}\cdot 0 - \frac{1}{3}(r_0 - r') = \frac{1}{6}(r_0 - r') \geq 0 \;,$$

as required.

Increase of Φ ($\Delta\Phi > \frac{1}{9}$): Here we show that in each round $r \in (r_0, r_f]$ the potential function increases by at least $\frac{1}{9}$. We use the notation $\Delta f(r)$ to denote $f(r) - f(r-1)$. We distinguish between the case that one of the messages in round $r-1$ was corrupted, and the case that both round $r-1$ messages were not corrupted.

In the former, we bound $\Delta\Psi_A(r) \leq 1$ (and the same for Bob). This suffices, since recall that $E_0(r) \setminus E_0(r-1)$ consists of the corrupted messages of round $r-1$ and hence

$$\Delta\Phi(r) = \Delta|E_0(r)|' - \frac{1}{3}\Delta\Psi_A(r) - \frac{1}{3}\Delta\Psi_B(r) \geq 1 - \frac{1}{3}\cdot 1 - \frac{1}{3}\cdot 1 = \frac{1}{3} \; > \frac{1}{9} \;.$$

We next bound $\Delta\Psi_A(r)$. We distinguish between three cases:

1. $S_{A,r-1} \in \{\text{Simulation}, \text{Verification}\}$: In this case $\Psi_A(r-1) = 0$. Moreover, $S_{A,r} \in \{\text{Simulation}, \text{Verification}, r\}$ and thus Equation (15) implies that $\Psi_A(r) = 0$. Hence, $\Delta\Psi_A(r) = 0 - 0 \leq 1$, as desired.

2. $S_{A,r-1} = r'$ and $S_{A,r} \in \{\text{Simulation}, \text{Verification}, r\}$: In this case, Equations (14) and (15) imply that $\Delta\Psi_A(r) = 0 - \Psi_A(r-1) \leq 0$, as desired.

3. $S_{A,r-1} = S_{A,r} = r'$: In this case, $v^0_{A,r} \geq v^0_{A,r-1}$ (since in round r, v^0 was not set to zero). Thus,

$$\Delta\Psi_A(r) = \left(r - r' - \frac{4}{3}v^0_{A,r}\right) - \left(r - 1 - r' - \frac{4}{3}v^0_{A,r-1}\right) = 1 - \frac{4}{3}(v^0_{A,r} - v^0_{A,r-1}) \leq 1 \ .$$

Since $S_{A,r} \in \{\text{Simulation}, \text{Verification}, r, S_{A,r-1}\}$, the above includes all possible cases. Thus we conclude that $\Delta\Psi_A(r) \leq 1$, which in turn implies that in this case, $\Delta\Phi(r) \geq \frac{1}{9}$.

Next we consider the second case, where there were no corrupted messages in round $r-1$. First, since $r \in C$ and C is not a Good Correction chunk it cannot be that $S_{A,r} = S_{B,r} = r$. Note that if $S_{A,r} = r$ then by Equations (14) and (15),

$$\Delta\Psi_A(r) = \Psi_A(r) - \Psi_A(r-1) = 0 - \Psi_A(r-1) \leq 0 \ .$$

We next show that if $S_{A,r} \neq r$ then $\Delta\Psi_A(r) \leq -\frac{1}{3}$ (and similarly for Bob), which will imply that

$$\Delta\Phi(r) = \Delta|E_0(r)| - \frac{1}{3}\Delta\Psi_A(r) - \frac{1}{3}\Delta\Psi_B(r) \geq 0 - \frac{1}{3}\cdot 0 - \frac{1}{3}\cdot\left(-\frac{1}{3}\right) = \frac{1}{9} \ ,$$

as required, where the second equation follows from the fact that it cannot be that $S_{A,r} = S_{B,r} = r$, and hence for at most one party Ψ equals 0, which implies that for at least one party Ψ is at most $-\frac{1}{3}$.

To this end, suppose that $S_{A,r} \neq r$. We argue that in the absence of error in round $r-1$, in the r'th round Alice will detect an inconsistency in their states. This is the case, since recall that we are in a Bad Correction chunk, and hence the states of Alice and Bob are inconsistent. Moreover, if Bob is in Correction state then he will send a message with a hash, and thus Alice will notice this inconsistency, and if Bob is not in Correction state and sends a short message, then Alice will notice this inconsistency since she is in Correction state, and thus expects longer message. Hence, it must be the case that $S_{A,r} \notin \{\text{Simulation}, \text{Verification}, r\}$.

This, together with the fact that $S_{A,r} \in \{\text{Simulation}, \text{Verification}, r, S_{A,r-1}\}$, implies that $S_{A,r} = S_{A,r-1} = r'$ for some $r' \leq r-1$. Since $m_{B,r-1}$ was not corrupted, Alice increases v^0 and hence $v^0_{A,r} = v^0_{A,r-1} + 1$. Therefore,

$$\Delta\Psi_A(r) = \left(r - r' - \frac{4}{3}v^0_{A,r}\right) - \left(r - 1 - r' - \frac{4}{3}v^0_{A,r-1}\right) = 1 - \frac{4}{3}(v^0_{A,r} - v^0_{A,r-1}) = -\frac{1}{3} \ ,$$

as required.

We thus conclude that

$$|C|' \leq 20|E_0|', \tag{16}$$

as desired. □

Lemma 34. *The total number of messages in all of the Good* Correction *chunks is bounded by* $11d|E|'$.

Proof. We partition the Good Correction chunks into two types: the *corrupted* Good Correction chunks, which have at least $\frac{1}{16}$ fraction of corrupted messages, and the *uncorrupted* Good Correction chunks, which have less than $\frac{1}{16}$ fraction of corrupted messages. By definition, the total number of messages in the corrupted Good Correction chunks is bounded by $16|E|'$. Thus, we need to show that the total number of messages in uncorrupted Good Correction chunks is at most $(11d-16)|E|'$

Let B be the set of all Bad chunks, Bad Correction chunks, and corrupted Good Correction chunks. We bound the total number of messages in the uncorrupted Good Correction chunks by $2|B|'+2|E|'$. This suffices since by Lemmas 32 and 33,

$$2|B|'+2|E|' \leq 2\left(5d|E|'+20|E|'+16|E|'\right)+2|E|' \leq (11d-16)|E|'\,,$$

where the last inequality follows from the fact that $d > 20$, which in turn follows from the definition of d and from the assumption that $\alpha < 0.01$ (see Equation (3)).

To this end, fix an uncorrupted Good Correction chunk C, let $r_0 \in C$ be its first round and let $r_f \in C$ be its last round. Let E_0 be the set of corrupted messages in all the chunks starting after the previous uncorrupted Good Correction chunk until (and including) chunk C. Note that $|E_0|' \geq 1$ since if after an uncorrupted Good Correction chunk there are no errors, then afterwards the parties will enter a Good Verification chunk followed by a Good Simulation chunk, and will never enter a Good Correction chunk again.[16]

Let $B_0 \subseteq B$ be the set of all chunks in B that came before C and after the previous uncorrupted Good Correction chunk, after the previous Good Verification chunk, and after the previous Good Simulation chunk. Note that B_0 may be empty. We prove that

$$|C|' \leq 2|B_0|'+2|E_0|'.$$

Note that this inequality holds trivially in the case where C consists of a single round. From now on, we assume that C has at least two rounds. Given two transcripts T_1 and T_2, let $T_1 \sqcap T_2$ be the longest shared prefix between T_1 and T_2. We define the disagreement between T_1 and T_2 by

$$T_1 \Delta T_2 = (T_1 \setminus (T_1 \sqcap T_2)) \circ (T_2 \setminus (T_1 \sqcap T_2))\,,$$

where $\circ$ denote the concatenation of the two strings.

We defined D_r to be the disagreement between the parties in round r. Namely,

$$D_r = (T_{A,r}[R_{A,r}]) \,\Delta\, (T_{B,r}[R_{B,r}])\,.$$

We next prove that

$$\frac{1}{2}|C|'-1 \leq |D_{r_0-1}|' \leq |B_0|' \tag{17}$$

This implies that

$$|C|' \leq 2|B_0|'+2 \leq 2|B_0|'+2|E_0|'\,,$$

as desired.

In the proof of Equation (17), we use the following two claims.

Claim 35. *The chunk immediately after C is a Good* Verification *chunk.*

Claim 36. *For any $i \in \{1,2\}$, in the presence of $< \frac{1}{4}(r_f - r_0)$ errors, the following holds,*

$$v^i_{A,r_f} > \frac{1}{4}(r_f-r_0) \iff T_A[R_A^{(i)}] \in \left\{T_B\left[R_B^{(1)}\right], T_B\left[R_B^{(2)}\right]\right\}.$$

[16] Note that this is not true for a corrupted Good Correction chunk, since after a corrupted Good Correction chunk the parties can immediately enter a Good Correction chunk, without any errors incurring in between.

Remark 37. *Recall that each round consists of two messages, and hence the fact that there are at most 1/16 corrupted messages in this chunk implies that there are at most 1/8 corrupted rounds in this chunk. We prove Claim 36 assuming that the fraction of corrupted round is less then 1/4. This stronger statement is needed later in the proof.*

Proof of Claim 36. Assume $v^i_{A,r_f} > \frac{1}{4}(r_f - r_0)$. By the definition of the protocol, this implies that the number of rounds between round $r_0 + \frac{r_f - r_0}{2}$ and round r_f, in which

$$H^{\mathsf{sent}}(T_A[R_A^{(i)}]) \in \left\{ H^{\mathsf{receive}}(T_B[R_B^{(1)}]), H^{\mathsf{receive}}(T_B[R_B^{(2)}]) \right\}$$

is more than $\frac{1}{4}(r_f - r_0)$. Recall that we are in the Ideal Hash Model, where a hash collision is thought of as an error. Hence, it must be the case that indeed

$$T_A[R_A^{(i)}] \in \{T_B[R_B^{(1)}], T_B[R_B^{(2)}]\}.$$

Next assume that $T_A[R_A^{(i)}] \in \{T_B[R_B^{(1)}], T_B[R_B^{(2)}]\}$. In this case, v^i increases by one at any round between $r_0 + \frac{r_f - r_0}{2}$ and r_f which had no error. Since as explained above, less than $\frac{1}{4}(r_f - r_0)$ rounds out of these $\frac{1}{2}(r_f - r_0)$ rounds may have errors, we conclude that $v^i_{A,r_f} > \frac{1}{4}(r_f - r_0)$, as desired.

□

Proof of Claim 35. Recall that each party in a Correction state, maintains variables v^0, v^1, v^2, where at each round, the party, say Alice, increases the variables v^i (for $i \in \{1, 2\}$) if and only if

$$H^{\mathsf{sent}}(T_A[R_A^{(i)}]) \in \left\{ H^{\mathsf{receive}}(T_B[R_B^{(1)}]), H^{\mathsf{receive}}(T_B[R_B^{(2)}]) \right\}.$$

Moreover, recall that for every $k \in \mathbb{N}$, whenever the party is in this Correction state for 2^k rounds, it sets $v^0 = v^1 = v^2 = 0$, and w is increased by a factor of 2, which redefines $R^{(1)}$ and $R^{(2)}$. For any round r, the value of $v^i_{A,r}$ in round r is updated in Step 3a and is then set to zero if and only if the party is in the Correction state for 2^k rounds for some $k \in \mathbb{N}$.

In what follows, we define $v^i_{A,r}$ to be the value of $v^i_{A,r}$ immediately after Step 3a (before it may have been set to zero), and we define the values $T_A\left[R_A^{(1)}\right], T_A\left[R_A^{(2)}\right], T_B\left[R_B^{(1)}\right], T_B\left[R_B^{(2)}\right]$ to be the values as defined in the second half of chunk C. (Recall that these values are being updated whenever the party is in this Correction chunk for 2^k rounds for some $k \in \mathbb{N}$. Suppose the party was in this Correction chunk for a total of 2^k rounds, then the values above are the values after the update that happened after 2^{k-1} rounds.)

We now prove that the chunk after chunk C is a Good Verification chunk. To this end, note that chunk C ends when one party, say Alice, changes her state from $S_{r_f - 1} = r_0$ to $S_{r_f} \neq r_0$. Hence, $r_f - r_0$ must be a power of two. Since in a Good Correction chunk, v^0 can increase only due to an error we get that $v^0_{A,r_f} \leq \frac{1}{4}(r_f - r_0)$ and hence Alice cannot change her state in Step 3c. We show by case analysis that $S_{A,r_f} = S_{B,r_f} =$ Verification, and at the end of round r_f it holds that $T_A[R_A] = T_B[R_B]$, which implies that $D_{r_f+1} = \emptyset$.

1. If $T_A[R_A^{(1)}] = T_B[R_B^{(1)}]$ then $v^1_{A,r_f}, v^1_{B,r_f} > \frac{1}{4}(r_f - r_0)$. Therefore, the parties define $R_A = R_A^{(1)}, R_B = R_B^{(1)}$ and hence

$$T_A[R_A] = T_A[R_A^{(1)}] = T_B[R_B^{(1)}] = T_B[R_B] \,.$$

2. If $T_A[R_A^{(2)}] = T_B[R_B^{(2)}]$ and $T_A[R_A^{(1)}] \neq T_B[R_B^{(1)}]$. Then $v^1_{A,r_f}, v^1_{B,r_f} \leq \frac{1}{4}(r_f - r_0)$ and $v^2_{A,r_f}, v^2_{B,r_f} > \frac{1}{4}(r_f - r_0)$. Therefore, the parties define $R_A = R_A^{(2)}, R_B = R_B^{(2)}$ and hence
$$T_A[R_A] = T_A[R_A^{(2)}] = T_B[R_B^{(2)}] = T_B[R_B] \ .$$

3. If $T_A[R_A^{(1)}] = T_B[R_B^{(2)}]$ then $T_B[R_B^{(1)}] \notin \{T_A[R_A^{(1)}], T_A[R_A^{(2)}]\}$. Therefore, $v^1_{A,r_f}, v^2_{B,r_f} > \frac{1}{4}(r_f - r_0)$ and $v^1_{B,r_f} \leq \frac{1}{4}(r_f - r_0)$. Thus, the parties define $R_A = R_A^{(1)}, R_B = R_B^{(2)}$ and hence
$$T_A[R_A] = T_A[R_A^{(1)}] = T_B[R_B^{(2)}] = T_B[R_B] \ .$$

4. The case $T_A[R_A^{(2)}] = T_B[R_B^{(1)}]$ is similar to the previous case.

5. If $\{T_A[R_A^{(1)}], T_A[R_A^{(2)}]\}$ and $\{T_B[R_B^{(1)}], T_B[R_B^{(2)}]\}$ do not intersect, then $v^1_{A,r_f}, v^2_{A,r_f}, v^1_{B,r_f}, v^2_{B,r_f} \leq \frac{1}{4}(r_f - r_0)$ and thus $S_{A,r_f} = S_{B,r_f} = r_0$, in contradiction to the fact that C ends in round r_f.

Moreover, since in each of the cases above, both parties set $S_{A,r_f} = S_{B,r_f} =$ Verification, at the end of round r_f the protocol enters a Good Verification chunk. □

We are now ready to prove Equation (17).

Upper bound of D_{r_0-1}. We show that $|D_{r_0-1}|' \leq |B_0|'$.

First suppose that $B_0 = \emptyset$. In this case the chunk preceding C is either a Good Verification chunk, a Good Simulation chunk, or an uncorrupted Good Correction chunk. However, as we saw, it cannot be the latter since after an uncorrupted Good Correction chunk must come a Good Verification chunk. Therefore, the chunk preceding C must be either a Good Verification chunk or a Good Simulation chunk, which implies that in round $r_0 - 1$ there is no disagreement on the transcript, and hence $|D_{r_0-1}|' = 0$.

We next assume that $B_0 \neq \emptyset$, and let $r' < r_0$ be the first round of B_0. Let $\{r_{A,i}\}_{i=0}^{k_A}$ be a sequence of rounds such that $r' - 1 = r_{A,0} < r_{A,1} < \ldots < r_{A,k_A} = r_0$, and
$$[r' - 1, r_0 - 1] = \bigcup_{i=0}^{k_A} [r_{A,i}, r_{A,i+1}),$$
where each $[r_{A,i}, r_{A,i+1})$ is either a single round r in which $S_{A,r} \in \{\text{Simulation}, \text{Verification}\}$, or a sequence of rounds r in which $S_{A,r} = r_{A,i}$ (this sequence may be of size 1). We next show
$$\begin{aligned} |D_{r_0-1}|' \ &\leq \ \left|T_{A,r'-1}[R_{A,r'-1}]\Delta T_{B,r'-1}[R_{B,r'-1}]\right|' \\ &+ \ \sum_{i=1}^{k_A} \left|T_{A,r_{A,i}}[R_{A,r_{A,i}}]\Delta T_{A,r_{A,i-1}}[R_{A,r_{A,i-1}}]\right|' \\ &+ \ \sum_{i=1}^{k_B} \left|T_{B,r_{B,i}}[R_{B,r_{B,i}}]\Delta T_{B,r_{B,i-1}}[R_{B,r_{B,i-1}}]\right|' \ . \end{aligned} \tag{18}$$
Then we will bound the number of messages in each of these terms in order to bound $|D_{r_0-1}|'$.

To prove equation (18) it suffices to prove that for any set of transcripts $\{T_i\}_{i=1}^{k}$,
$$|T_1 \Delta T_k|' \leq \sum_{i=1}^{k-1} |T_i \Delta T_{i+1}|'. \tag{19}$$

To this end, it suffices to show that $|T_1 \Delta T_2|' \leq |T_1 \Delta T_3|' + |T_2 \Delta T_3|'$. Indeed,

$|T_1 \Delta T_2|' =$
$|T_1 \setminus (T_1 \sqcap T_2)|' + |T_2 \setminus (T_1 \sqcap T_2)|' \leq$
$|T_1 \setminus (T_1 \sqcap T_3)|' + |(T_1 \sqcap T_3) \setminus (T_1 \sqcap T_2 \sqcap T_3)|' + |T_2 \setminus (T_2 \sqcap T_3)|' + |(T_2 \sqcap T_3) \setminus (T_1 \sqcap T_2 \sqcap T_3)| \leq$
$|T_1 \setminus (T_1 \sqcap T_3)|' + |T_3 \setminus (T_2 \sqcap T_3)|' + |T_2 \setminus (T_2 \sqcap T_3)|' + |T_3 \setminus (T_1 \sqcap T_3)| =$
$|T_1 \Delta T_3|' + |T_2 \Delta T_3|'$,

as desired.

We show that

$$| \left(T_{A,r'-1}[R_{A,r'-1}] \Delta T_{B,r'-1}[R_{B,r'-1}]\right) |' = 0 \ .$$

By Claim 35, after an uncorrupted Good Correction chunk, the parties enter a Good Verification chunk, which is not in B. Thus, round $r'-1$ cannot be in a uncorrupted Good Correction chunk and therefore must be in a Good Verification or a Good Simulation chunk. Either way, in round $r'-1$ the parties agree on their transcripts and thus

$$\left(T_{A,r'-1}[R_{A,r'-1}] \Delta T_{B,r'-1}[R_{B,r'-1}]\right) = \emptyset,$$

as required.

We show that for any $i \in \{1, \ldots, k_A\}$ we have that

$$|T_{A,r_{A,i}}[R_{A,r_{A,i}}] \Delta T_{A,r_{A,i-1}}[R_{A,r_{A,i-1}}]|' \leq r_{A,i} - r_{A,i-1} \ . \tag{20}$$

First consider the case where $[r_{A,i-1}, r_{A,i})$ contains only one round r and

$$S_{A,r} \in \{\text{Simulation}, \text{Verification}\}.$$

If Alice detects an inconsistency in round r then $T_{A,r+1}[R_{A,r+1}] = T_{A,r}[R_{A,r}]$. Otherwise, $T_{A,r+1}[R_{A,r+1}] = (T_{A,r}[R_{A,r}], m)$ for some message m. Either way,

$$|T_{A,r_{A,i}}[R_{A,r_{A,i}}] \Delta T_{A,r_{A,i-1}}[R_{A,r_{A,i-1}}]|' = |T_{A,r+1}[R_{A,r+1}] \Delta T_{A,r}[R_{A,r}]|' \leq 1 = r_{A,i} - r_{A,i-1} \ .$$

Now consider the case where $[r_{A,i-1}, r_{A,i})$ contains all rounds r in which $S_{A,r} = r_{A,i-1}$. In this case $S_{A,r_{A,i}} \in \{\text{Verification}, r_{A,i}\}$. If $S_{A,r_{A,i}} = r_{A,i}$, then $T_{A,r_{A,i}}[R_{A,r_{A,i}}] = T_{A,r_{A,i-1}}[R_{A,r_{A,i-1}}]$. If $S_{A,r_{A,i}} = \text{Verification}$, then $T_{A,r_{A,i}} = T_{A,r_{A,i-1}}$ and $R_{A,r_{A,i}} \in \{R^{(1)}, R^{(2)}\}$, as defined in the second half of the rounds $[r_{A,i-1}, r_{A,i})$. In either case,

$$|T_{A,r_{A,i}}[R_{A,r_{A,i}}] \Delta T_{A,r_{A,i-1}}[R_{A,r_{A,i-1}}]|' \leq R_{A,r_{A,i-1}} - R^{(2)} \leq 2w_{A,r_{A,i}-1} = 2 \cdot \frac{1}{2}(r_{A,i} - r_{A,i-1}) = r_{A,i} - r_{A,i-1} \ .$$

This proves Equation (20). This, together with Equation (18), implies that

$$\begin{aligned}
|D_{r_0-1}|' \ \leq \ & \left|T_{A,r'-1}[R_{A,r'-1}] \Delta T_{B,r'-1}[R_{B,r'-1}]\right|' + \\
& \sum_{i=1}^{k_A} \left|T_{A,r_{A,i}}[R_{A,r_{A,i}}] \Delta T_{A,r_{A,i-1}}[R_{A,r_{A,i-1}}]\right|' + \\
& \sum_{i=1}^{k_B} \left|T_{B,r_{B,i}}[R_{B,r_{B,i}}] \Delta T_{B,r_{B,i-1}}[R_{B,r_{B,i-1}}]\right|' \leq \\
& 0 + \sum_{i=1}^{k_A} (r_{A,i} - r_{A,i-1}) + \sum_{i=1}^{k_B} (r_{B,i} - r_{B,i-1}) = 2(r_0 - r') = |B_0|' \ ,
\end{aligned}$$

as desired.

Lower bound on the $|D_{r_0-1}|$. Recall that we need to prove that $|D_{r_0-1}|' \geq \frac{1}{2}|C|' - 1$. We prove this by proving that

$$|D_{r_0-1}| \geq \frac{1}{2}(r_f - r_0).$$

Note that at least one party changes its state in round r_f, hence $r_f - r_0$ must be a power of two. Let $r' = r_0 + \frac{r_f - r_0}{2}$ be the previous round in C that was a power of two. We show that $D_{r'} = D_{r_0-1}$. This is done by observing that for any $r \in [r_0 - 1, r']$ we have that

$$T_{A,r} = T_{A,r+1} \ , \ T_{B,r} = T_{B,r+1} \ , \ R_{A,r} = R_{A,r+1} \ , \ R_{B,r} = R_{B,r+1}.$$

Indeed, this is true for $r = r_0 - 1$, since in this case the state is updated to be $S_{r_0} = r_0$ and none of the variables above are changed. Moreover, these values are not changed at any round r of the correction state, except the last round.

Since in round r' the parties did not change their state, $v^2_{A,r'} \leq \frac{1}{4}(r' - r_0)$. By Claim 36, we have that $T_{A,r'}\left[R^{(2)}_{A,r'}\right] \neq T_{B,r'}\left[R^{(2)}_{B,r'}\right]$. Thus $T_{A,r'}[R_{A,r'}] \sqcap T_{B,r'}[R_{B,r'}]$ does not include round $R^{(2)}_A = R^{(2)}_B$, and therefore,

$$\begin{aligned}
|D_{r_0-1}|' &= |D_{r'}|' \\
&= |T_{A,r'}[R_{A,r'}] \Delta T_{B,r'}[R_{B,r'}]|' \\
&\geq (R_{A,r'} - R^{(2)}_{A,r'}) + (R_{B,r'} - R^{(2)}_{B,r'}) \\
&\geq w_{A,r'} + w_{B,r'} \\
&\geq \frac{1}{2}(r_f - r_0) \ ,
\end{aligned}$$

as required.

□

Lemma 38. *The total number of messages in all Good* Verification *chunks is bounded by* $300d\left(\log \frac{1}{\beta}\right)|E|'$.

Proof. We partition all the rounds in all Good Verification chunks into 3 (non-consecutive) parts.

- P_1 consists of all the rounds r in Good Verification chunks that satisfy: $R_{A,r} = R_{A,r-1}$ and $T_{A,r-1}[R_{A,r-1} + 1] = T_{B,r-1}[R_{B,r-1} + 1]$.
- P_2 consists of all the rounds r in Good Verification chunks that satisfy: $R_{A,r} = R_{A,r-1} + 1$.
- P_3 consists of all the rounds r in Good Verification chunks that satisfy: $R_{A,r} = R_{A,r-1}$ and $T_{A,r-1}[R_{A,r-1} + 1] \neq T_{B,r-1}[R_{B,r-1} + 1]$.

We note that a more natural order of this partition would have been P_1, P_3, P_2. However, in the analysis we bound $|P_3|'$ as a function of $|P_1|'$ and $|P_2|'$, and hence the unnatural order. Note that these three parts cover all the Good Verification chunks since by the definition of the protocol, for any round r in these chunks, we have that $R_{A,r} \in \{R_{A,r-1}, R_{A,r-1} + 1\}$. Note that $R_{A,r+1}$ is always defined since the protocol ends only when Alice is in a Simulation state. We now bound the number of messages in each of these parts.

First part: We show that

$$|P_1|' \leq 2|E|'. \tag{21}$$

To get the bound on $|P_1|'$ we show that for every round $r \in P_1$, $m_{B,r}$ is corrupted.[17] By the definition of the protocol, if $S_{A,r-1} = \text{Verification}$ then $R_{A,r} = R_{A,r}$ if and only if the hash that Alice receives indicates that $T_{B,r-1}[R_{B,r-1}+1] \neq T_{A,r-1}[R_{A,r-1}+1]$. Since, by our assumption, these transcripts are equal, it must be the case that $m_{B,r}$ is corrupted.

Second part: We show that

$$|P_2|' \leq 17d|E|'. \tag{22}$$

Let B be the set of all Bad chunks, Bad Correction chunks, and Good Correction chunks. By Lemmas 32, 33, and 34, it holds that

$$|B|' \leq 5d|E|' + 20|E|' + 11d|E|' \leq 17d|E|',$$

where the latter inequality follows from the fact that $d \geq 100$, which in turn follows from the fact that $\alpha \leq 0.01$.

Thus, to prove Equation (22), it suffices to prove that

$$|P_2|' \leq |B|'.$$

To this end, let d_r be the difference between the number of rounds in $T_{A,r}$ and $R_{A,r}$, i.e., $d_r = \frac{1}{2}|T_{A,r}|' - R_{A,r}$.[18] Let r_f be the last round in the protocol. Let $\{r_{A,i}\}_{i=0}^{k}$ be a sequence of rounds such that $0 = r_{A,0} < r_{A,1} < \ldots < r_{A,k_A} = r_f$, and

$$[0, r_f - 1] = \bigcup_{i=0}^{k_A - 1} [r_{A,i}, r_{A,i+1}),$$

where each $[r_{A,i}, r_{A,i+1})$ is either a single round r in which $S_{A,r} \in \{\text{Simulation}, \text{Verification}\}$, or a sequence of all rounds r for which $S_{A,r} = r_{A,i}$.[19] Note that all the intervals in which Alice is in a Correction state are contained in B. By definition of the protocol, d_r has the following properties:

1. If $[r_{A,i}, r_{A,i+1})$ consists of a single round r in which $S_{A,r} = \text{Simulation}$ then $d_{r_{A,i+1}} - d_{r_{A,i}} = 0$.
2. If $[r_{A,i}, r_{A,i+1})$ consists of a single round r in which $S_{A,r} = \text{Verification}$ then $d_{r_{A,i+1}} - d_{r_{A,i}} \in \{0, -1\}$.

   Moreover, for any round in $r \in P_2$ we have that $d_{r+1} - d_r = -1$.
3. If $[r_{A,i}, r_{A,i+1})$ consists of all round r in which $S_{A,r} = r_{A,i}$ then $d_{r_{A,i+1}} - d_{r_{A,i}} \leq r_{A,i+1} - r_{A,i}$.
4. $d_0 = 0$, and for every r it holds that $d_r \geq 0$.

[17] Recall that $|E|'$ denotes the number of messages that have been corrupted, whereas $|P_1|'$ denotes the number of messages sent in the rounds of P_1, which is *twice* the number of rounds in P_1, since in each round two messages are sent, one by Alice and one by Bob.

[18] Note that d_r is unrelated to the constant d.

[19] A similar partition was considered in the proof of Lemma 34.

Thus,

$$\begin{aligned}
0 &\leq d_{r_f} \\
&= d_{r_f} - d_0 \\
&= \sum_{i=0}^{k} \left(d_{r_{A,i+1}} - d_{r_{A,i}}\right) \\
&\leq \sum_{i:S_{r_{A,i}}=r_{A,i}} (r_{A,i+1} - r_{A,i}) - \sum_{i:r_{A,i}\in P_2} (-1) \\
&\leq \frac{1}{2}|B|' - \frac{1}{2}|P_2 \cap [0, r_f - 1]|' \\
&= \frac{1}{2}|B|' - \frac{1}{2}|P_2|' ,
\end{aligned}$$

where the last equality follows from the fact that $r_f \notin P_2$. Hence, $|P_2|' \leq |B|'$ as required.

Third phase: Let B be the set of all BAD chunks, Good and Bad Correction chunks, and all the messages in P_1 and P_2. In other words, B consists of all the messages, except those in P_3 and those that belong to a Good Simulation. Lemmas 32, 33, and 34, together with Equations (21) and (22), imply that

$$|B|' \leq 35d|E|'.$$

We prove that

$$|P_3|' \leq (7|B|' + 18|E|') \log \frac{1}{\beta}. \tag{23}$$

which together with the above, implies that

$$|P_3|' \leq 246d|E|' \log \frac{1}{\beta}. \tag{24}$$

Equations (21) (22) and (24), imply that

$$|P_1|' + |P_2|' + |P_3|' \leq 18d|E|' + 281d|E|' \log \frac{1}{\beta} < 300d|E|' \log \frac{1}{\beta},$$

as desired.

It thus remains to prove Equation (23). To this end, consider all the rounds which are in Good Simulation chunks. Denote these rounds by $1 = r_0 < r_1 < \ldots < r_t$. We divide the protocol into chunks $C_1, \ldots, C_t$ where $C_i = [r_{i-1}, r_i]$.[20] We denote by $B_i = B \cap C_i$, $E_i = E \cap C_i$, and $P_{3,i} = P_3 \cap C_i$. We prove that for every $i \in [t]$,

$$|P_{3,i}|' \leq (7|B_i|' + 9|E_i|') \log \frac{1}{\beta}. \tag{25}$$

Note that this implies Equation (23) since all the B_i's are disjoint (follows from the fact that all the r_i's belong to Good Simulation chucks), and hence $|B|' = \sum_{i=1}^{t} |B_i|'$. Moreover, $\sum_{i=1}^{t} |E_i|' \leq 2|E|'$ since each message in E belongs to at most two E_i's.

We next prove Equation (25) via the use of a potential function. In what follows, we focus on a specific chunk C_i, but we omit the subscript i from the notations to avoid cluttering.

[20]Note that these chunks are not disjoint, however, they do cover all the rounds in the protocol.

Moreover, we abuse notation, and denote by r_0 the first round in this chunk and by r_f the last round in the chunk. We denote by B_r all the messages in B from round r_0 to round r (including). We denote by $\text{sim}_A(r)$ the largest round $\leq r$ such that $S_{A,r-1} = \text{Simulation}$. We denote by

$$P_{3,A} = \left\{m_r \in P_3 : |T_{A,r}| - |T_{A,r}[R_{A,r}]| \leq \ell^{\mathsf{sent}}_{A,r}\right\}$$

We define $P_{3,B}$ analogously, and we note that

$$|P_3|' \leq |P_{3,A}|' + |P_{3,B}|' + |E|'.$$

Thus, it remains to prove that

$$|P_{3,A}|', |P_{3,B}|' \leq (3.5|B_i|' + 4|E_i|') \log \frac{1}{\beta}.$$

We focus on bounding $|P_{3,A}|'$. Bounding $|P_{3,B}|'$ is done analogously. For every round $r \in [r_0, r_f]$ we define the potential function:

$$\Phi_A(r) =$$
$$3|B_r|' \log \frac{1}{\beta} + \log \tilde{\ell}_{A,r} - |(\text{sim}_A(r), r] \setminus P_{3,A}| - \left(\frac{1}{2}|T_{A,r}|' - R_{A,r}\right) \cdot \log \frac{1}{\beta},$$

where $\tilde{\ell}_{A,r}$ is defined as follows: If r is such that $S_{A,r} = \text{Simulation}$ then $\tilde{\ell}_{A,r} = \ell^{\mathsf{sent}}_{A,r}$. Else, if $\ell^{\mathsf{sent}}_{A,r} = \ell^{-}_{A,r-1}$ then we define $\tilde{\ell}_{A,r} = \max\{\beta^{-1}, \alpha\tilde{\ell}_{A,r-1}\}$. Else, $\ell^{\mathsf{sent}}_{A,r} = \ell^{+}_{A,r-1}$ in which case we define $\tilde{\ell}_{A,r} = 2\tilde{\ell}_{A,r-1}$.

The fact that round r_0 and r_f belong to a Good Simulation chunk, implies that

$$\Phi_A(r_0) = \log \tilde{\ell}_{A,r_0},$$

and

$$\Phi_A(r_f) \leq 3|B|' \log \frac{1}{\beta} + \log \tilde{\ell}_{A,r_f}.$$

Thus,

$$\begin{aligned}
&\Phi_A(r_f) - \Phi_A(r_0) = \\
&3|B|' \log \frac{1}{\beta} + \log \tilde{\ell}_{A,r_f} - \log \tilde{\ell}_{A,r_0} = \\
&3|B|' \log \frac{1}{\beta} + \log \frac{\tilde{\ell}_{A,r_f}}{\tilde{\ell}_{A,r_0}} \leq && (26)\\
&3|B|' \log \frac{1}{\beta} + \log \beta^{-\frac{1}{2}|T_{A,r_0} \Delta T_{A,r_f}|' - 2} \leq && (27)\\
&3|B|' \log \frac{1}{\beta} + \log \beta^{-\frac{1}{2}|B|' - 4} = \\
&(3.5|B|' + 4) \log \frac{1}{\beta}\,,
\end{aligned}$$

where recall that $T_1 \Delta T_2 = (T_1 \setminus T_1 \sqcap T_2) \cup (T_2 \setminus T_1 \sqcap T_2)$. Equation 26 follows from the (α, β)-smoothness of T_{A,r_0} and T_{A,r_f} as follows: let ℓ be the length of the last message in $T_{A,r_0} \sqcap T_{A,r_f}$. Than Equation 26 follows from combining the follows,

$$\ell_{A,r_f} \leq \beta^{-(\frac{1}{2}|T_{A,r_0} \setminus T_{A,r_0} \sqcap |T_{A,r_f}|' + 1)} \ell$$

and

$$\ell_{A,r_0} \geq \alpha^{\frac{1}{2}|T_{A,r_0} \setminus T_{A,r_0} \sqcap |T_{A,r_f}|'+1} \ell \geq \beta^{\frac{1}{2}|T_{A,r_0} \setminus T_{A,r_0} \sqcap |T_{A,r_f}|'+1} \ell \ .$$

To prove Equation 27, since r_0 and r_f in good simulation, it will be suffice to show that $|T_{A,r_0}[R_{A,r_0}]\Delta T_{A,r_f}[R_{A,r_f}]|' \leq |B|'$. This is done in a similar way to the proof of Equation 20. We will partition the regine $[r_0, r_f]$ into regines $[r_{A,i}, r_{A,i+1})$ such that each regime contain a single round such that $S_{A,r_{A,i}} \in \{\text{Simulation}, \text{Verification}\}$ or all rounds r such that $S_{A,r_{A,i}} = r_{A,i}$. By the protocol, if $[r_{A,i-1}, r_{A,i})$ contain a single round of Verification than $T_{A,r_{A,i-1}}[R_{A,r_{A,i-1}}]\Delta T_{A,r_{A,i}}[R_{A,r_{A,i}}] = \emptyset$. Else, if $[r_{A,i}, r_{A,i+1})$ contain a single round of Simulation than two messages, or none, are added to $T[R]$ and so $|T_{A,r_{A,i}}[R_{A,r_{A,i}}]\Delta T_{A,r_{A,i+1}}[R_{A,r_{A,i+1}}]|' \leq 2$. Else, where $[r_{A,i-1}, r_{A,i})$ contain that all round in which $S_{A,r} = r_{A,i-1}$ we have that R can decreased by at most $2w_{r_{A,i}-1} = r_{A,i} - r_{A,i-1}$. Thus, using Equation 19 we have,

$$\begin{aligned} |T_{A,r_0}[R_{A,r_0}]\Delta T_{A,r_f}[R_{A,r_f}]|' &\leq \sum_{i=1}^{k-1} |T_{A,r_{A,i}}[R_{A,r_{A,i}}]\Delta T_{A,r_{A,i+1}}[R_{A,r_{A,i+1}}]|' \\ &\leq 2\,|\{r \in [r_0, r_f) \mid S_{A,r} \neq \text{Verification}\}| \leq |B|' - 4 \ . \end{aligned}$$

We partition $[r_0, r_f)$ into chunks $(r_{A,i-1}, r_{A,i}]$ such that each chunk consists with a single round r where $S_{A,r-1} \in \{\text{Simulation}, \text{Verification}\}$, or all consecutive rounds, where $S_{A,r-1} = r_i$ for some $r' \in \mathbb{N}$.

We now note that for every i such that $(r_{A,i-1}, r_{A,i}]$ consists with a single round in $P_{3,A}$,

$$\Phi_A(r_{A,i}) - \Phi_A(r_{A,i-1}) = \Phi_A(r) - \Phi_A(r-1) \geq 1. \tag{28}$$

This follows from the fact that the only term in the potential function Φ_A that changes from round r to round $r+1$ is $\tilde{\ell}_{A,r}$, and by the definition of P_3 it holds that $\tilde{\ell}_{A,r-1} \geq 2\tilde{\ell}_{A,r}$, which implies Equation (28).

It remains to argue that for all other i,

$$\Phi_A(r_{A,i}) - \Phi_A(r_{A,i-1}) \geq 0. \tag{29}$$

We consider the following four cases:

Case 1: $(r_{A,i-1}, r_{A,i}]$ consists with all rounds r in which $S_{A,r-1} = r_{i-1}$. In this case note that $R_{r_{A,i}} \geq R_{r_{A,i-1}} - 2(r_{A,i} - r_{A,i-1})$. Moreover, since all the messages in this chunk are of the form ℓ^-_{r-1}, we have that $\tilde{\ell}_{A,r_i} \geq \alpha^{r_{A,i}-r_{A,i-1}} \tilde{\ell}_{A,r_{A,i-1}}$ and thus

$$\Phi_A(r_{A,i}) - \Phi_A(r_{A,i-1}) \geq 6(r_{A,i} - r_{A,i-1}) \log \frac{1}{\beta} - (r_i - r_{i-1}) \log \frac{1}{\alpha} - (r_{A,i} - r_{A,i-1}) - 2(r_{A,i} - r_{A,i-1}) \log \frac{1}{\beta} \geq 0 \ .$$

Case 2: The i-th chunk consists with a single round $r \in P_3 \setminus P_{3,A}$ and $S_{A,r} \neq$ Simulation. In this case, it is easy to see that

$$\Phi_A(r) - \Phi_A(r-1) = \log \tilde{\ell}_{A,r} - \log \tilde{\ell}_{A,r-1} - 1 \geq 0.$$

Case 3: The i-th chunk consist with a single round $r \in B$ and $(S_{A,r-1}, S_{A,r}) \neq$ (Verification, Simulation) In this case,

$$\begin{aligned} &\Phi_A(r) - \Phi_A(r-1) \geq \\ &6 \log \frac{1}{\beta} + \log \tilde{\ell}_{A,r} - \log \tilde{\ell}_{A,r-1} - 1 \geq \\ &6 \log \frac{1}{\beta} - \log \frac{1}{\alpha} - 1 \geq 0 \ . \end{aligned}$$

where the second inequality follows from the fact that when Alice does not move from Verification to Simulation, $\tilde{\ell}$ can drop by no more than a factor of α (Claim 30).

Case 4: The i-th regime consists with a single round $r \in B$ such that $S_{A,r-1} =$ Verification and $S_{A,r} =$ Simulation. By definition of Φ,

$$\Phi_A(r) - \Phi_A(r-1) \geq$$
$$6\log\frac{1}{\beta} + \log\tilde{\ell}_{A,r} - \log\tilde{\ell}_{A,r-1} + |(\mathrm{sim}_A(r-1), r-1] \setminus P_{3,A}| + \left(\frac{1}{2}|T_{A,r-1}|' - R_{A,r-1}\right) \cdot \log\frac{1}{\beta}$$

Thus, it suffices to show that

$$\log\tilde{\ell}_{A,r} \geq \log\tilde{\ell}_{A,r-1} - |(\mathrm{sim}_A(r-1), r-1] \setminus P_{3,A}| - \left(\frac{1}{2}|T_{A,r-1}|' - R_{A,r-1}\right) \cdot \log\frac{1}{\beta} - 6\log\frac{1}{\beta} \quad (30)$$

Claim 39. *For every r, the following holds:*

$$\log\tilde{\ell}_{A,r} - |(\mathrm{sim}_A(r), r] \setminus P_{3,A}| \leq \log|T_{A,r}[R_{A,r}, \infty]| + \log\frac{1}{\beta}$$

Proof. For $r \in P_{3,A}$ the claim holds trivially, since by definition of $P_{3,A}$,

$$\tilde{\ell}_{A,r} \leq \ell^{\mathsf{sent}}_{A,r} \leq 2|T_{A,r}[R_{A,r}, \infty]|$$

and hence

$$\log\tilde{\ell}_{A,r} \leq \log|T_{A,r}[R_{A,r}, \infty]| + 1,$$

as desired.

For r such that $S_{A,r} \in$ Simulation, it holds that

$$\tilde{\ell}_{A,r} \leq \ell^{\mathsf{sent}}_{A,r} \leq \beta^{-1}|T_{A,r}[R_{A,r}, \infty]|,$$

where the latter follows from the smoothness condition. This implies that

$$\log\tilde{\ell}_{A,r} \leq \log\frac{1}{\beta} + \log|T_{A,r}[R_{A,r}, \infty]|,$$

as desired.

Next we prove the rest of the claim by induction on r. Suppose the claim is true for round r, and we prove that it is true for round $r+1$ for which $S_{A,r+1} \notin \{\text{Simulation}, P_{3,A}\}$. We distinguish between the case that $R_{A,r+1} \leq R_{A,r}$ and the case where $R_{A,r+1} = R_{A,r} + 1$. In the former case, the fact that $S_{A,r+1} \notin$ Simulation, implies that $\tilde{\ell}_{A,r+1} \leq 2\tilde{\ell}_{A,r}$, and hence by our induction hypothesis,

$$\begin{aligned}
&\log\tilde{\ell}_{A,r+1} \leq \\
&\log\tilde{\ell}_{A,r} + 1 \leq \\
&\log|T_{A,r}[R_{A,r}, \infty]| + \log\frac{1}{\beta} + |(\mathrm{sim}_A(r), r] \setminus P_{3,A}| + 1 \leq \\
&\log|T_{A,r+1}[R_{A,r+1}, \infty]| + \log\frac{1}{\beta} + |(\mathrm{sim}_A(r), r] \setminus P_{3,A}| + 1 = \\
&\log|T_{A,r+1}[R_{A,r+1}, \infty]| + \log\frac{1}{\beta} + |(\mathrm{sim}_A(r+1), r+1] \setminus P_{3,A}|,
\end{aligned}$$

as desired.

We next consider the case where $R_{A,r+1} = R_{A,r} + 1$. In this case $\tilde{\ell}_{A,r+1} = \max\{\beta^{-1}, \alpha \cdot \tilde{\ell}_{A,r}\}$. If $\tilde{\ell}_{A,r+1} = \beta^{-1}$ then the claim holds trivially. On the other hand, if $\tilde{\ell}_{A,r+1} = \alpha \cdot \tilde{\ell}_{A,r}$ then by our induction hypothesis,

$$\begin{aligned}
&\log \tilde{\ell}_{A,r+1} = \\
&\log \alpha \cdot \tilde{\ell}_{A,r} = \\
&\log \alpha + \log \tilde{\ell}_{A,r} \leq \\
&\log \alpha + \log |T_{A,r}[R_{A,r}, \infty]| + \log \frac{1}{\beta} + |(\mathrm{sim}_A(r), r] \setminus P_{3,A}| \leq \\
&\log \alpha + \log \left(\frac{1}{\alpha} \cdot |T_{A,r+1}[R_{A,r+1}, \infty]| \right) + \log \frac{1}{\beta} + |(\mathrm{sim}_A(r), r] \setminus P_{3,A}| \leq \\
&\log |T_{A,r+1}[R_{A,r+1}, \infty]| + \log \frac{1}{\beta} + |(\mathrm{sim}_A(r+1), r+1] \setminus P_{3,A}|,
\end{aligned}$$

as desired.

□

We now prove Equation (30) using Claim 39.

$$\begin{aligned}
&\log \tilde{\ell}_{A,r-1} - |(\mathrm{sim}_A(r-1), (r-1)] \setminus P_{3,A}| - \left(\frac{1}{2} |T_{A,r-1}|' - R_{A,r-1} \right) \cdot \log \frac{1}{\beta} - 6 \log \frac{1}{\beta} \leq \\
&\log |T_{A,r-1}[R_{A,r-1}, \infty]| - \left(\frac{1}{2} |T_{A,r-1}|' - R_{A,r-1} \right) \cdot \log \frac{1}{\beta} - 5 \log \frac{1}{\beta} \,.
\end{aligned}$$

Thus, we need to prove that

$$\log \tilde{\ell}_{A,r} \geq \log |T_{A,r-1}[R_{A,r-1}, \infty]| - \left(\frac{1}{2} |T_{A,r-1}|' - R_{A,r-1} \right) \cdot \log \frac{1}{\beta} - 2 \log \frac{1}{\beta}.$$

To this end, let $k \triangleq \frac{1}{2} |T_{A,r-1}|' - R_{A,r-1}$, and let ℓ denote the length of the longest message in round $R_{A,r-1}$ of in the transcript $T_{A,r-1}$. The fact that $S_{A,r} = \text{Simulation}$, together with the smoothness property of the original protocol, implies that

$$\tilde{\ell}_{A,r} = \ell^{\mathsf{sent}}_{A,r} \geq \alpha \ell \geq \ell \cdot \beta,$$

and hence

$$\log \tilde{\ell}_{A,r} \geq \log \ell - \log \frac{1}{\beta}. \tag{31}$$

On the other hand, the smoothness property implies that

$$|T_{A,r-1}[R_{A,r-1}, \infty]| \leq \sum_{i=0}^{k} 2 \frac{\ell}{\beta^i} \leq \frac{2\ell}{\beta^k (1-\beta)} \leq \frac{\ell}{\beta^{k+1}},$$

which in turn implies that

$$\log |T_{A,r-1}[R_{A,r-1}, \infty]| \leq \log \ell + (k+1) \cdot \log \frac{1}{\beta}.$$

Therefore,

$$\log|T_{A,r-1}[R_{A,r-1},\infty]| - \left(\frac{1}{2}|T_{A,r-1}|' - R_{A,r-1}\right)\cdot\log\frac{1}{\beta} - 5\log\frac{1}{\beta} \leq$$
$$\log\ell + (k+1)\cdot\log\frac{1}{\beta} - \left(\frac{1}{2}|T_{A,r-1}|' - R_{A,r-1}\right)\cdot\log\frac{1}{\beta} - 5\log\frac{1}{\beta} =$$
$$\log\ell + (k+1)\cdot\log\frac{1}{\beta} - k\cdot\log\frac{1}{\beta} - 5\log\frac{1}{\beta} =$$
$$\log\ell - 4\log\frac{1}{\beta} \leq$$
$$\log\tilde{\ell}_{A,r+1}$$

as desired, where the latter inequality follows from Equation (31).

The result follows.

□

Lemma 40. *The total volume of all Good* Verification *chunks, Good* Correction *chunks, and Bad* Correction *chunks is at most* $4\beta^{-1}|E| + 35d\beta^{-1}|E|'$

Proof. We partition the rounds in Good Verification chunks into two part. The first part consists of all the rounds that satisfy

$$T_{A,r-1}[R_{A,r-1}+1] \neq T_{B,r-1}[R_{B,r-1}+1]$$

and the second part consists of all the rounds that satisfy

$$T_{A,r-1}[R_{A,r-1}+1] = T_{B,r-1}[R_{B,r-1}+1].$$

We bound the first part of Good Verification by $2|E|$ and the rest of those chunks by $(4\beta^{-1} - 2)|E| + 35d\beta^{-1}|E|'$.

Bounding the volume of all the rounds in the second part of Good Verification**.** We remain consistent with the notations we used in the proof of Lemma 38, and denote all the messages that belong to the second part of Good Verification by P_3. We prove that $|P_3| \leq 2|E|$.

To this end, fix any rounds r_0, r_f such that both r_0 and r_f belong to a Good Simulation chunk, and all the rounds $r \in (r_0, r_f)$ do not belong to a Good Simulation chunk. Let $C_1, ..., C_k$ be all the (maximal) sets of consecutive rounds in $P_3 \cap (r_0, r_f)$ such that there are errors only on the last message in each chunk, i.e. we start a new chunk after a corrupted message or whenever the rounds stop being consecutive) . Let C_i^+ to be C_i together with the round before it and let $E_0 = E \cap (r_0, r_f)$. We show that $|C_i| \leq 4|C_i^+ \cap E_0|$ for every $i \neq k$, and $|C_k| \leq 2|E_0 \setminus \bigcup_{i=1}^{k-1} C_i|$. This yield a total bound of $\sum_{i=1}^{k} |C_i| \leq 8|E_0|$, and therefor $|P_3| \leq 8|E|$.

We start with the former. Fix any $i \in \{1, \ldots, k-1\}$. Since after chunk C_i the parties do not enter a Good Simulation chunk it must be the case that C_i ended with an error, as otherwise, the parties would have doubled their message size until they had enough budget to erase the inconsistency in their transcripts, and would have entered a Simulation state. Denote by r the first round in C_i and denote by $r + c$ the last round in C_i. The fact that chunk C_i is in P_3, and have errors only in the last round, implies that for every $i \in \{1, \ldots, c\}$ it holds that $\ell_{\max,r+i} \geq 2 \cdot \ell_{\max,r+i-1}$. Hence,

$$|C_i| \leq \ell_{\max,r+c}\left(1 + \frac{1}{2} + \left(\frac{1}{2}\right)^2 + \ldots + \left(\frac{1}{2}\right)^c\right) \leq 2\ell_{\max,r+c}.$$

Thus, where both parties send messages of the same length in round $r+c$, since the round $r+c$ is corrupted, we have that,

$$|C_i| \leq 2|C_i^+ \cap E_0|.$$

In the other case, the parties disagree on $\ell_{\max,r+c-1}$, and thus the longer message in round $r+c-1$ must be corrupted. In this case we get that $\ell_{\max,r+c} \leq 2\ell_{\max,r+c-1}$ and we get,

$$|C_i| \leq 4|C_i^+ \cap E_0|,$$

as desired.

We next prove that $|C_k| \leq 2|E_0 \setminus \bigcup_{i=1}^{k-1} C_i|$. To this end, denoting by r the first round in C_k, note that

$$|C_k| \leq \max\{|T_{A,r}[R_{A,r},\infty]|, |T_{A,r}[R_{B,r},\infty]|\} + |E_0 \setminus \bigcup_{i=1}^{k-1} C_i|,$$

where $R_{A,r} = R_{B,r}$ and $T_{A,r}[R_{A,r}] = T_{B,r}[R_{B,r}]$ since we are in a Good Verification chunk, and $T_{A,r}[R_{A,r}+1] \neq T_{B,r}[R_{B,r}+1]$ since C_k is in P_3. Therefore, all the messages in $T_{A,r}[R_{A,r},\infty]$ and $T_{B,r}[R_{B,r},\infty]$ were added due to error. These messages were added in $E_0 \setminus \bigcup_{i=1}^{k-1} C_i$ since before E_0 was a Good Simulation chunk and in rounds $\bigcup_{i=1}^{k-1} C_i$ the transcripts do not change. Therefore,

$$|C_k| \leq \max\{|T_{A,r}[R_{A,r},\infty]|, |T_{A,r}[R_{B,r},\infty]|\} + |E_0 \setminus \bigcup_{i=1}^{k-1} C_i| \leq 2|E_0 \setminus \bigcup_{i=1}^{k-1} C_i|,$$

as desired.

The rest of the chunks Let B be the set of all Bad chunks, and let G be the set of all Bad Correction chunks, Good Correction chunks, and the first part of Good Verification chunks (which consists only of messages in $P_1 \cup P_2$). We bound

$$|G| \leq \beta^{-1}|G|' + \left(3\beta^{-1}+3\right)|E| + 3\alpha|B| + 35d\beta^{-1}|E|' .$$

By Lemmas 32, 33, 34, and the proof of Lemma 38, this is the desired bound, since:

$$\begin{aligned}
|G| &\leq \beta^{-1}|G|' + \left(3\beta^{-1}+3\right)|E| + 3\alpha|B| + 6d\beta^{-1}|E|' \\
&\leq \beta^{-1}\left(20|E|' + 11d|E|' + 2|E|' + 17d|E|'\right) + \left(3\beta^{-1}+3\right)|E| \\
&\quad +3\alpha\left((2\beta^{-1}+1)|E| + 3d\beta^{-1}|E|'\right) + 6d\beta^{-1}|E|' \\
&\leq \left(3\beta^{-1}+3+3\alpha(2\beta^{-1}+1)\right)|E| + 35d\beta^{-1}|E|' \\
&\leq (4\beta^{-1}-8)|E| + 35d\beta^{-1}|E|'
\end{aligned}$$

where the last two inequalities follow from the choice of our parameters (see Equation (3)).

Let C be a set of consecutive rounds in G. We define $B_0 = \emptyset$ if the chunk preceding C is not a bad chunk, and otherwise we define B_0 to be the bad chunk preceding C. Let E_0 be the set of corrupted messages in C and in the chunks preceding C until, not included, the previous part of G. We show that

$$|C| \leq \beta^{-1}|C|' + \left(3\beta^{-1}+3\right)|E_0| + 3\alpha|B_0| + 3d\beta^{-1} .$$

To prove that this suffices, we bound the number of sets of consecutive rounds in G, by $|E|'+1$. This is done as follows: Denote by $G_1, \ldots, G_t$ the (ordered) set of all consecutive

rounds in G, where G_1 is the first set of consecutive rounds and G_t is the last. We show that for every $i \in \{1, \ldots, t-1\}$, there must exist an error in the interval between (including) the first round of G_i and (excluding) the first round of G_{i+1}. Indeed, suppose (for contradiction) that this interval had no error. If G_i started with a Bad Correction chunk, then the protocol would have moved into a Good Correction chunk, and from there into (either the first part or the second part of) a Good Verification chunk. If G_i started with any other chunk, then it will also move into (either the first part or the second part of) a Good Verification. In either case, after being in the first part of a Good Verification chunk, the protocol will move into a Good Simulation chunk or to the second part of a Good Verification chunk. However, after being in the second part of the Good Verification chunk, in the absence of error, the protocol will always move and remain in a Good Simulation chunk until the end, in contradiction to the existence of G_{i+1}.

We show that in each round of C, except the first one, a player that did not receive a corrupted message in round $r-1$ sends a message of length ℓ^-_{r-1} in round r (i.e., Alice sends a message of length $\ell^-_{A,r-1}$ and Bob sends a message of length $\ell^-_{B,r-1}$).

- If r is in a Good Correction chunk then each player indeed sends a message of this length.
- If r is in a Bad Correction chunk then one player, say Alice, is in a Correction state in round r , and thus sends a message of length $\ell^-_{A,r-1}$. We prove the condition for Bob by case analysis on the chunk at round $r-1$.

  If in round $r-1$ the parties are in a Bad Correction chunk, then without an error Bob detects the inconsistency, moves to a Correction state and sends a message of length $\ell^-_{B,r-1}$.

  If in round $r-1$ Bob was in a Good Correction state, then also in round r he will send a message of length $\ell^-_{B,r-1}$.

  If in round $r-1$ the parties were in the second part of a Verification chunk, then Bob's message in round $r-1$ is corrupted.
- If r is in Good Verification , then both players are in a Verification state, $T_{A,r-1}[R_{A,r-1}] = T_{B,r-1}[R_{B,r-1}]$ and $T_{A,r-1}[R_{A,r-1}+1] = T_{B,r-1}[R_{B,r-1}+1]$. By the definition of the protocol, without corruption, both players send a message of length ℓ^-_{r-1} in round r.

Thus, by Claim 31,

$$|C| \leq \beta^{-1}|C|' + \left(3\beta^{-1}+3\right)|E_0 \cap C| + 3\ell^{\mathsf{sent}}_{\max,r_0}, \tag{32}$$

where r_0 be the first round of C.

We next show that,

$$\ell^{\mathsf{sent}}_{\max,r_0} \leq \beta^{-1}|E_0 \setminus C| + \alpha|B_0| + d\beta^{-1} \ .$$

Which implies that $|C| \leq \beta^{-1}|C|' + \left(3\beta^{-1}+3\right)|E_0| + 3\alpha|B_0| + 3d\beta^{-1}$, as required.

We distinguish between two cases: The case that in round $r_0 - 1$ one of the messages was corrupted or one of the messages sent was shorter than d, and the case that both $r_0 - 1$ round messages are of length $\geq d$ and not corrupted.

In the former case, the smoothness guaranty (Claim 30) implies that

$$\ell^{\mathsf{sent}}_{\max,r_0} \leq \beta^{-1}\min\{\ell^{\mathsf{sent}}_{A,r_0-1}, \ell^{\mathsf{receive}}_{B,r_0-1}\} \leq \beta^{-1}\max\{|E_0 \setminus C|, d\} = \beta^{-1}|E_0 \setminus C| + d\beta^{-1} \ ,$$

where the second to last equation follow from the fact that one of the messages in round $r_0 - 1$ is corrupted or has length $< d$.

We next consider the latter case, where there were no corrupted messages in round $r_0 - 1$ and both of the messages have length $\geq d$. In this case we show that the chunk preceding C must be a Bad chunk. This is done by a process of elimination: It cannot be in first part of Good Verification or Good Simulation chunk, since if in round $r_0 - 1$ the parties agree on their state and transcript, in the absence of corruptions, the parties continue agree on their state and transcript and the protocol will remain in part 2 of Good Verification or move to Good Simulation chunk. It cannot be the first part of Good Verification chunk, Good Correction chunk or Bad Correction chunk, since in this case, by the definition of C, it will be unite with C.

We now show that C cannot start with a Bad Correction chunk. Assume toward contradiction that it does start with a Bad Correction chunk. Then one party, say Alice, did not set her state $S_{A,r_0} = r_0$, and hence did not detect an inconsistency. This implies that the message she received m_{B,r_0-1} was not sent with a hash or was corrupted, in contradiction.

Since r_0 in Good Correction chunk or second part of Good Verification chunk. Thus

$$\ell^{\mathsf{sent}}_{\max,r_0} = \ell^{-}_{r_0-1} \leq \max\{\alpha|B_0|, \beta^{-1}\} \ ,$$

where the last equality follows from the fact that $r_0 - 1$ is in a Bad chunk.

□

Combining Lemmas 32, 33, 34, 38, and 40, we obtain Lemma 29. We next prove Theorem 9 given Lemma 29.

Proof of Theorem 9. First we note that Item 5 follows immediately from the efficient nature of the protocol.

To prove Item 1 note that a player, say Alice, aborts if the underlying protocol Π instructs her to abort given the (partial) transcript she is holding, denoted by $T_{A,r}$. In this case, $|T_{A,r}| \geq t_{\min}$ (by definition). It remains to note that for every round r, the communication complexity of $\Pi'_{\mathcal{A}}$ up until round r is at least $|T_{A,r}|$. This follows from the fact that in the protocol Π', Alice increases $T_{A,r}$ only in Step 1d, where she adds $(m_{A,r-1}, m_{B,r-1})$ after sending the message $m_{A,r-1}$ and receiving the message $m_{B,r-1}$.

To prove Items 2, 3 and 4 we rely on the following claim.

Claim 41.

$$|T_{A,r_f} \sqcap T_{B,r_f}| \geq |G| - |B| \quad \textit{and} \quad |T_{A,r_f} \sqcap T_{B,r_f}|' \geq |G|' - 2|B|' \ ,$$

where G is the set of all rounds without errors in Good Simulation chunks, B is the rest of the rounds, and r_f be the last round of the protocol. Moreover, recall that $T_1 \sqcap T_2$ is the longest shared prefix between T_1 and T_2.

We defer the proof of Claim 41, and first show why this claim implies Items 2, 3 and 4. First, we bound $|B|$ and $|B|'$. To do so we bound the number and volume of messages in rounds with errors in Good Simulation chunks. Note that in any such round, if a message of length ℓ was corrupted, by the smoothness property of the original protocol, there is at most one uncorrupted message of length $\leq \beta^{-1}\ell$ in this round. Summing over all such rounds, we get that there are at most $2e'$ messages in corrupted rounds in Good Simulation chunks, and the volume of these messages is at most $(1 + \beta^{-1})e$. Combine it with Lemma 29, we get

$$|B| \leq 9\beta^{-1}e + 10d\beta^{-1}e' \quad \text{and} \quad |B|' \leq 302d \log \frac{1}{\beta} e' \ .$$

To prove Item 2, note that by Claim 41,

$$\mathrm{CC}(\Pi) \geq |T_{A,r_f} \sqcap T_{B,r_f}| \geq |G| - |B| \ .$$

This holds since the partial transcript that both parties agree on must be consistent with Π. Thus,

$$\mathrm{CC}(\Pi'_{\mathcal{A}}) = |G| + |B| \leq \mathrm{CC}(\Pi) + 2|B| \leq \mathrm{CC}(\Pi) + 18\beta^{-1}e + 20d\beta^{-1}e' \ .$$

To prove Item 3 note that by Claim 41,

$$R(\Pi) \geq |T_{A,r_f} \sqcap T_{B,r_f}|' \geq |G|' - 2|B|' \ ,$$

and thus,

$$R(\Pi'_{\mathcal{A}}) = |G|' + |B|' \leq R(\Pi) + 3|B|' \leq R(\Pi) + 906d \log \frac{1}{\beta} e' \ .$$

Finally, to prove Item 4 it remains to note that by the proof of Item 1,

$$|T_{A,r_f}|, |T_{B,r_f}| \leq \mathrm{CC}(\Pi'_{\mathcal{A}}) \ ,$$

and by Claim 41,

$$|T_{A,r_f} \sqcap T_{B,r_f}| \geq |G| - |B| = \mathrm{CC}(\Pi'_{\mathcal{A}}) - 2|B| = \mathrm{CC}(\Pi'_{\mathcal{A}}) - 18\beta^{-1}e - 20d\beta^{-1}e' \ .$$

Proof of Claim 41. The proof uses the potential functions

$$\Phi(r) \triangleq |T_{A,r+1} \sqcap T_{B,r+1}|$$

and

$$\Phi'(r) \triangleq |T_{A,r+1}[R_{A,r+1}] \sqcap T_{B,r+1}[R_{B,r+1}]|' - 2\tilde{w}_{A,r} - 2\tilde{w}_{B,r} \ ,$$

where $\tilde{w}_{A,r} = r - S_{A,r}$ if $S_{A,r} \in \mathbb{N}$, and is zero otherwise ($\tilde{w}_{B,r}$ is defined analogously). We prove that

$$\Phi(r_f) \geq |G| - |B| \quad \text{and} \quad \Phi'(r_f) \geq |G|' - 2|B|' \ .$$

As before we denote $\Delta\Phi(r) = \Phi(r) - \Phi(r-1)$ and $\Delta\Phi'(r) = \Phi'(r) - \Phi'(r-1)$.

Since $T_{A,0}$ and $T_{B,0}$ ar empty, we get that

$$\Phi(0) = 0 \quad \text{and} \quad \Phi'(0) = 0 \ .$$

In any round $r \in G$ both parties are in Simulation (and so $T[R] = T$) and agree on T and R. Thus they both increase the agreement on the transcript by the messages $m_{A,r}, m_{B,r}$. Together with the fact that in this case also w_A, w_B remain 0, we get,

$$\Delta\Phi(r) = \ell_{A,r} + \ell_{B,r} \quad \text{and} \quad \Delta\Phi'(r) = 2 \ .$$

Hence, the total contribution of all round in G for Φ is $|G|$ and for Φ' is $|G|'$. By definition,

$$\Delta\Phi(r) \geq -\max\{|T_{A,r-1} - T_{A,r}|, |T_{B,r-1} - T_{B,r}|\} \ .$$

Assume without lost of generality that $|T_{A,r-1} - T_{A,r}| \geq |T_{B,r-1} - T_{B,r}|$. Note that $|T_{A,r-1} - T_{A,r}|$ can be positive only when in round r Alice execute Step 2b. In this case both the message that she send and received in the round r was of size larger that what she erased. Thus we get that for $r \in B$,

$$\Delta\Phi(r) \geq -\ell_{A,r} - \ell_{B,r} \ ,$$

and so the total contribution of all the rounds in B to Φ is at least $-|B|$.

Note that the only place in the protocol where $|T_A[R_A] \sqcap T_B[R_B]|'$ can decrease is only where $|T_A[R_A]|'$ or $|T_B[R_B]|'$ decreased, which is in Steps 3d or 3e. In both cases we get that it decreased by at most $2w_A$ (res. $2w_B$) and $\tilde{w}_A$ (res. $\tilde{w}_B$) decreased to 0. Thus, since $2\tilde{w}_A > w_A$ (rep. $2\tilde{w}_B > w_B$) we get that in total execute Steps 3d or 3e does not decrease Φ'. Thus, Φ' can decrease only where $\tilde{w_A}$ or $\tilde{w_B}$ increased, which by definition it can be by only 1 per round. Thus for any round $r \in B$ we get that

$$\Delta\Phi'(r) \geq -4 \,.$$

Thus the total contribution of all the rounds in B is at least $2|B|$. By concluding the total contributions of the round from G and B to Φ and Φ' we get that

$$\Phi(r_f) \geq |G| - |B| \quad \text{and} \quad \Phi'(r_f) \geq |G|' - 2|B|' \,.$$

□

□

C Proofs from Section 5

C.1 Proof of Lemma 15

We partition E into $E_1, ..., E_5, \tilde{E}_1, ..., \tilde{E}_5$ where for every $k \in [5]$, E_k is the set of messages with hash collisions on Z_k, and $\tilde{E}_k$ is the set of messages with hash collisions on the function H'_k, but not on Z_k. Lemma 15 follows from the next 4 claims.

Claim 42. $\forall k \in [5]$, $\Pr[|E_k| \leq 2\gamma t] \geq 1 - e^{-\frac{1}{6}t}$.

Proof. Recall that by definition in each round r,

$$Z_k = \begin{cases} (f^{2^{w_r}}_{x_{k,r}}(V_i), 0) & \text{if } |V_k| \geq 2^{w_r} \\ (V_k, 1) & \text{if } |V_k| < 2^{w_r} \end{cases}$$

Note that if $|V_k| < 2^{w_r}$ then there is no hash collision by definition. Next, consider the case that $|V_k| \geq 2^{w_r}$. In this case, by definition,

$$|Z_k| = 2^{w_r} = 2^{\lceil \alpha \ell_r \rceil + \lceil u_r \rceil + 9\lceil \log \frac{1}{\gamma} \rceil + 6},$$

which greater than $\gamma^{-1}\ell \log \frac{1}{\gamma}$.

We consider only the first $\gamma^{-1}\ell$ hash chunks of Z_k, each of size $\log \frac{1}{\gamma}$. The total number of such chunks (of Z_k) in the entire protocol is at most $\gamma^{-1}t$.

We first consider only *oblivious* adversaries, namely, ones that choose which bits to corrupt, and whether the corruption is a toggle, insert, or delete, independently of the common random string. Note that for any such oblivious adversary O, each hash chunk has a collision probability of γ. The next claim bounds the number of such adversaries.

Claim 43. *The number of oblivious adversaries that make at most ϵt errors in the first t bits of the protocol, is bounded by $2^{2\epsilon \log \frac{1}{\epsilon} t}$ for $\epsilon \leq 0.05$.*

Proof. Each oblivious adversary can be described as a set of at most ϵt elements, where each element in the set is a pair of the form (i, o) where $i \in [t]$ and o is one of the following operations: Toggle, Delete, Insert 0, Insert 1.

Each such set S corresponds to the following oblivious adversary: Let i be the smallest number for which there exists o such that $(i, o) \in S$. Corrupt the i'th bit that is sent as instructed by o, and remove the element (i, o) from S. Continue recursively, while considering the *updated* transcript (including the corruptions). Namely, in the recursion, when considering the smallest i for which $(i, o) \in S$ for some o, we corrupt the i'th bit that is sent relative to the (current) *corrupted* transcript. Namely, if the (current) corrupted transcript is of length k, we consider the next bit to be sent to be the $k + 1$'st bit, even though the parties may have tried to send many more (or less) bits, and a gap occurred due to many deletions (or insertions). In particular, the set S may include multiple elements of the form (i, o) where $o =$ Delete, and hence S is actually a multi-set.

Thus, the number of oblivious adversaries is bounded by,

$$\frac{t^{\epsilon t}}{(\epsilon t)!}5^{\epsilon t} \leq \frac{t^{\epsilon t}}{(\frac{\epsilon t}{e})^{\epsilon t}}5^{\epsilon t} = 2^{\log(\frac{5e}{\epsilon})\epsilon t} \leq 2^{2\log(\frac{1}{\epsilon})\epsilon t} .$$

□

By the Chernoff bound (Lemma 11), the probability that there are $> 2t$ hash chunks with a hash collision, is at most $e^{-\frac{1}{3}t}$. Note that a hash collision in a message of length ℓ corresponds to at least $\gamma^{-1}\ell$ chunks with hash collisions. Therefore, for any oblivious adversary O, it hold that

$$\Pr_x[|E_k| > 2\gamma t] \leq e^{-\frac{1}{3}t} .$$

Using union bound over all $\leq \binom{t}{\epsilon t} \cdot 4^{\epsilon t} \leq 2^{2\epsilon \log \frac{1}{\epsilon} t} \leq e^{\frac{1}{6}t}$ possible oblivious adversaries, we get that

$$\Pr_x[\forall O : |E_k| > 2\gamma t] \leq e^{-\frac{1}{6}t} .$$

The result follows. □

Claim 44. $\forall k \in [5]$, $\Pr[|\tilde{E}_k| < 2\gamma t] \geq 1 - e^{-\frac{\alpha\gamma}{3\log\frac{1}{\gamma}}t}$.

Proof. Consider the r-th message with hash, and assume $V_k^A \neq V_k^B$. This message has w_r bits of hash. Thus, by Lemma 14 under the uniform seed, it has a collision probability 2^{-w_r}, and hence under the 2^{-w_r}-biased distribution, it has a collision probability of at most

$$2 \cdot 2^{-w_r} \leq 2^{-\alpha\ell_r - \log\frac{1}{\gamma}} = \gamma^{\frac{1}{\log\frac{1}{\gamma}} \cdot (\alpha\ell_r + \log\frac{1}{\gamma})} \leq \gamma^{\left\lceil \frac{\alpha\ell_r}{\log\frac{1}{\gamma}} \right\rceil}.$$

So the probability of having such a hash collision in a message of length ℓ is at most the probability that $\left\lceil \frac{\alpha\ell}{\log\frac{1}{\gamma}} \right\rceil$ independent Bernoulli variables with probability γ are all one. For each $a \in [r]$, we denote by

$$n_a \triangleq \left\lceil \frac{\alpha\ell_a}{\log\frac{1}{\gamma}} \right\rceil ,$$

and denote these Bernoulli random variables by $X_{a,1}, ..., X_{a,n_r}$. Consider the set of all such variables $\{X_{a,i} \mid a \in [r], i \in [n_a]\}$.

Since each message of length ℓ contributes at least $\frac{\alpha\ell}{\log\frac{1}{\gamma}}$, their total number is at least $\frac{\alpha}{\log\frac{1}{\gamma}}t$. By the Chernoff bound (Lemma 11), with probability $\geq 1 - e^{-\frac{\alpha\gamma}{3\log\frac{1}{\gamma}}t}$ less then $\frac{2\alpha\gamma}{\log\frac{1}{\gamma}}t$ of them are 1. In this case,

$$\begin{aligned}|\tilde{E}_k| \leq & \sum_{a\in[r]:X_{a,1}=\ldots=X_{a,n_a}=1} \ell_a \leq \\ & \sum_{a\in[r]:X_{a,1}=\ldots=X_{a,n_a}=1} n_a \cdot \frac{\ell_a}{n_a} \leq \\ & \sum_{a\in[r]:X_{a,1}=\ldots=X_{a,n_a}=1} n_a \cdot \max_{a\in[r]}\left\{\frac{\ell_a}{n_a}\right\} \leq \\ & |\{i,j \mid X_{i,j}=1\}| \cdot \frac{\log\frac{1}{\gamma}}{\alpha} \leq \\ & \frac{2\alpha\gamma}{\log\frac{1}{\gamma}}t \cdot \frac{\log\frac{1}{\gamma}}{\alpha} = 2\gamma t.\end{aligned}$$

□

Claim 45. $\forall k \in [5]$, $\Pr[|\tilde{E}_k|' \geq 4\gamma r] \leq e^{-\frac{2}{3}\gamma r}$

Proof. Consider a round with hash such that $V_k^A \neq V_k^B$. Note that for a uniformly distributed $U \in \{0,1\}^{\frac{1}{\gamma}}$,

$$\Pr[f_U^w(V_k^A) = f_U^w(V_k^B)] = 2^{-w} \leq \gamma.$$

Since $|S| = 2C \cdot w > 2C \cdot \log\frac{1}{\gamma}$ we have that $G(S)$ is γ-biased. Therefore, by Lemma 14,

$$\Pr[f_{G(S)}^w(V_k^A) = f_{G(S)}^w(V_k^B)] \leq 2\gamma.$$

This, together with the Chernoff bound (Lemma 11), implies that the probability that there are more than $4\gamma r$ hash collisions is at most $e^{-\frac{2}{3}\gamma r}$, as desired. □

Claim 46. $\forall k \in [5]$, $\Pr\left[|E_k|' \geq 10\gamma r\right] \leq 2 \cdot 2^{-4r}$

Proof. Let E_k^A be the messages with hash collisions on Z_k that Alice sends. We will show that

$$\Pr\left[|E_k^A|' \geq 5\gamma r\right] \leq 2^{-4r}.$$

A similar equation holds also for Bob, and the result follows.

Given a set of massages received by Alice, which we denote by $T = (m_{B,1}^{\mathsf{receive}}, \ldots, m_{B,r}^{\mathsf{receive}})$, we denote a partition of its rounds $\{1,\ldots,r\}$ by $p(T) = (r_1, r_2, ..., r_k)$ according to the following process: Let $r_0 = 0$. Suppose r_i is already defined, we will define r_{i+1} as follows:

We first define for every $r \in Q \setminus [r_i]$,

$$U_r = \frac{t_r - t_{r_i}}{a_r - a_{r_i}},$$

where t_r is the amount of communication Alice *received* until round r, and a_r is the number of rounds in which Alice *sent* a hash. In what follows, we abuse notation and often denote a_{r_i} by a_i. Similarly, we often denote t_{r_i} and U_{r_i}, by t_i and U_i, respectively.

We define

$$U'_{i+1} = \min_{r\in Q\setminus[r_i]} U_r.$$

We next define

$$r_{i+1} = \max\{r \in Q \setminus [r_i] | U_r < \gamma^{-6} U'_{i+1}\}. \tag{33}$$

We define $\Delta a_i = a_{r_{i+1}} - a_{r_i}$, which is the number of rounds with hash in the i'th regime. We define $\Delta t_i = t_{i+1} - t_i$, which is the volume of the i'th regime. We define

$$U_i \triangleq \frac{\Delta t_i}{\Delta a_i}, \tag{34}$$

which intuitively, would be the average message length if we glued together all consecutive messages without a hash. Thus,

$$U'_i \leq U_i \leq \gamma^{-6} U'_i. \tag{35}$$

Moreover, we argue that

$$U'_{i+1} \geq \gamma^{-6} U'_i. \tag{36}$$

To this end, let $r > r_i$ be the round that satisfy $U'_{i+1} \triangleq \frac{t_r - t_i}{a_r - a_i}$. Suppose for the sake of contradiction that $U'_{i+1} = \frac{t_r - t_i}{a_r - a_i} < \gamma^{-6} U'_i$. This, together with the fact that $\frac{t_i - t_{i-1}}{a_i - a_{i-1}} < \gamma^{-6} U'_i$, implies that

$$\frac{t_r - t_{i-1}}{a_r - a_{i-1}} = \frac{a_r - a_i}{a_r - a_{i-1}} \cdot \frac{t_r - t_i}{a_r - a_i} + \frac{a_i - a_{i-1}}{a_r - a_{i-1}} \cdot \frac{t_i - t_{i-1}}{a_i - a_{i-1}} < \gamma^{-6} U'_i \,.$$

Since $\frac{t_r - t_{i-1}}{a_r - a_{i-1}} < \gamma^{-6} U'_i$, by Equation (33), we have that $r \leq r_i$, in contradiction to our choice of r.

We say that a regime $(r_i, r_{i+1}]$ is *heavy* if $\Delta t_i > \frac{1}{2}\gamma t_i$, and we say that is *light* otherwise. In what follows, we first argue that the total number of rounds in the light regimes is at most $2\gamma r$. We then argue that with probability at least $1 - 8r \cdot 2^{-8\gamma^2 r}$, the total number of rounds with hash collisions in all heavy regimes is at most $5\gamma r$. The result follows.

Light regimes. We first show that the total number of rounds with hashes in the light regimes is at most $2\gamma r$. To this end, first note that

$$\Delta t_i \leq \frac{1}{2}\gamma t_i = \frac{1}{2}\gamma \sum_{j=0}^{i-1} \Delta t_j.$$

Therefore, for any probability distribution $(p_0, \ldots, p_{i-1})$,[21] such that $p_j > \frac{1}{2^{i-j}}$ for every $j \in \{0, 1, \ldots, i-1\}$, it holds that

$$\Delta t_i \leq \sum_{j=0}^{i-1} p_j 2^{i-j-1} \gamma \Delta t_j = \mathbb{E}_{j \sim p}[2^{i-j-1} \gamma \Delta t_j].$$

Hence, there exists $j \in \{0, 1, \ldots, i-1\}$ that satisfies

$$\Delta t_i \leq 2^{i-j-1} \gamma \Delta t_j.$$

For such j, we say that regime i is *directly controlled* by regime j. Note that

$$U_i \geq U'_i \geq \left(\gamma^{-6}\right)^{i-j} U'_j \geq \left(\gamma^{-6}\right)^{i-j-1} U_j,$$

[21] Namely, $\sum_{j=0}^{i-1} p_j = 1$ and $p_j \geq 0$ for every $j \in \{0, 1, \ldots, i-1\}$.

where the first and third inequality follow from Equation (35), and the second inequality follows from Equation (36). Thus,

$$\Delta a_i = \frac{\Delta t_i}{U_i} \leq \frac{2^{i-j-1}\gamma \Delta t_j}{(\gamma^{-6})^{i-j-1} U_j} \leq \frac{2^{i-j-1}\gamma}{(2\gamma^{-5})^{i-j-1}} \cdot \frac{\Delta t_j}{U_j} = \gamma^{5(i-j)-1}\Delta a_j \leq \gamma^{i-j}\Delta a_j \ .$$

Moreover, note that if there exists a series of rounds $j = i_0, i_1, ..., i_m = i$, where each i_k is directly controlled by i_{k-1}, then

$$\Delta a_i \leq \gamma^{i-i_{m-1}}\gamma^{i_{m-1}-i_{m-2}} \cdots \gamma^{i_1-j}\Delta a_j = \gamma^{i-j}\Delta a_j.$$

In this case we say that regime i is *controlled* by regime j, and denote this by $j \prec i$.

Since, as we argued, every light regime i is (directly) controlled by some previous regime j, it follows that for every light regime i there exists a heavy regime j such that regime i is controlled by regime j. Hence,

$$\sum_{i \in \text{Light}} \Delta a_i \leq \sum_{j \in \text{Heavy}} \sum_{i : j \prec i} \Delta a_i \leq \sum_{j \in \text{Heavy}} \sum_{j \prec i} \gamma^{i-j}\Delta a_j = \sum_{j \in \text{Heavy}} \Delta a_j \sum_{i : j \prec i} \gamma^{i-j} \leq \frac{\gamma}{1-\gamma} \sum_{j \in \text{Heavy}} \Delta a_j \leq 2\gamma r,$$

as desired.

Heavy regimes Next we show that with probability $1 - 2^{-4r}$ the number of rounds with hash collisions in all the heavy regimes i is at most $3\gamma r$.

Consider all the regimes that satisfy $t_{i+1} < \gamma r$. In these regimes in total there are at most γr rounds with hash. Thus it suffice to show that w.h.p for all the heavy regimes with $t_{i+1} \geq \gamma r$ there is a total of at most $2\gamma r$ rounds with hash collisions.

To this end, we show that for each such heavy regime i, the probability that there are more than $2\gamma \Delta a_i$ hash collisions, is at most $4 \cdot 2^{-7r}$. Given this, the claim follows since $\sum \Delta a_i \leq r$ and $r \cdot 4 \cdot 2^{-7r} \leq 2^{-4r}$ together with a a straightforward union bound over all $\leq r$ heavy regimes.

We will use the following claim.

Claim 47. *For any adversary and heavy regimes between the t_i to t_{i+1} bits of the protocol, the probability that the regime has more than $2\gamma\Delta a$ rounds of hash collisions, is at most $4 \cdot 2^{-7\gamma^{-1}t_{i+1}}$.*

Since we consider only regimes where $t_{i+1} \geq \gamma r$, the result follows.

Proof of Claim 47. For each oblivious adversary, over the i-th regime, the number of hash bits of any message in this regime is

$$2^{w_r} \geq 2^{\max_{r' \in Q \cap [r-1]} \log \frac{t_r - t_{r'}}{a_r - a_{r'}} + 9 \log \frac{1}{\gamma} + 6} = 64\gamma^{-9} \max_{r' \in Q \cap [r-1]} \frac{t_r - t_{r'}}{a_r - a_{r'}} \geq 64\gamma^{-9} U_i' \geq 64\gamma^{-3} U_i \ ,$$

When the first inequality follows from the definition of w_r and the latter inequality follows from Equation (35). Thus, each message has at least $64\gamma^{-2}U_i$ hash blocks of size $\log \frac{1}{\gamma} \leq \frac{1}{\gamma}$. We consider only the first $64\gamma^{-2}U_i$ blocks for each message. The number of all such hash blocks is

$$64\gamma^{-2}U_i \Delta a_i = 64\gamma^{-2}\Delta t_i \geq \frac{64\gamma^{-2}t_{i+1}}{2\gamma^{-1}+1} \geq 24\gamma^{-2}t_{i+1} \ ,$$

where the second equation follows since in heavy regimes $t_{i+1} = t_i + \Delta t_i \leq (2\gamma + 1)\Delta t_i$. Since we consider an oblivious adversary, the probability of having a hash collision in each block is γ. Hence, by the Chernoff bound (Lemma 11), the probability that more than 2γ fraction of them have a hash collision, is at most $2^{-8\gamma^{-1}t_{i+1}}$. By union bound over all $\leq 2^{t_{i+1}}$ oblivious adversaries we get that with probability $\leq 2^{-(8\gamma^{-1}-1)t_{i+1}} \leq 2^{-7\gamma^{-1}t_{i+1}}$ there are less then $2\gamma\Delta a_i$ rounds with hash collisions. In particular, the result follows. □

□

C.2 Proof of Lemma 16.

In the proof of Lemma 16 we use the following claim.

Claim 48. *For any $R \in \mathbb{N}$ and any monotone increasing series $T = \{t_r\}_{r=1}^{R}$, consider the continuous version of T, defined by setting $t_0 = 0$, and for any $r \in \mathbb{N}$ and $\eta \in [0,1)$ setting $t_{r+\eta} = (1-\eta)t_r + \eta t_{r+1}$. Let*

$$S_T(a) = \int_0^a \max_{b<r} \left\{ \ln \frac{t_r - t_b}{r-b} \right\} dr \ .$$

Then

$$\forall a \in \mathbb{Z} : S_T(a) \leq e \cdot t_a,$$

where e is the base of the natural log.

The proof of this claim is deferred to Appendix C.3. In what follows we use Claim 48 to prove Lemma 16.

Proof of Lemma 16. In what follows, we use notions (such as u_r and t_r), defined in Section 5.2, with respect to the protocol $\Pi_{\mathcal{A}}^{\mathcal{H}}$.

To bound $\sum_{r \in Q^A} u_r$ define the series $T' = \{t'_i\}$ such that for any $r \in Q^A$ let $t'_{a_r^A} = t_r^A$. Namely, t'_i is the amount of communication (in bits) that Alice received until (and including) the round where she sent the i'th message with a hash.

$$
\begin{aligned}
\sum_{r\in Q^A} u_r &= \sum_{r\in Q^A} \max_{r'\in Q^A\cap[r-1]} \log \frac{t^A_r - t^A_{r'}}{a^A_r - a^A_{r'}} \\
&\le \sum_{r\in Q^A} \max_{r'\in Q^A\cap[r-1]} \log\left(2\frac{t^A_r - t^A_{r'}}{a^A_r + 1 - a^A_{r'}}\right) \\
&\le \sum_{r\in Q^A} \max_{r'\in Q^A\cap[r-1]} \int_{a=a^A_r}^{a^A_r+1} \log 2\frac{t'_a - t^A_{r'}}{a - a^A_{r'}} da \\
&= \sum_{r\in Q^A} \max_{r'\in Q^A\cap[r-1]} \int_{a=a^A_r}^{a^A_r+1} \log \frac{2}{\alpha}\frac{\alpha t'_a - \alpha t^A_{r'}}{a - a^A_{r'}} da \\
&= |Q^A|\log\frac{2}{\alpha} + \sum_{r\in Q^A} \max_{r'\in Q^A\cap[r-1]} \int_{a=a^A_r}^{a^A_r+1} \log \frac{\alpha t'_a - \alpha t^A_{r'}}{a - a^A_{r'}} da \\
&= |Q^A|\log\frac{2}{\alpha} + \frac{1}{\ln 2}\sum_{r\in Q^A} \max_{r'\in Q^A\cap[r-1]} \int_{a=a^A_r}^{a^A_r+1} \ln \frac{\alpha t'_a - \alpha t^A_{r'}}{a - a^A_{r'}} da \\
&\le |Q^A|\log\frac{2}{\alpha} + \frac{1}{\ln 2}\sum_{r\in Q^A} \int_{a=a^A_r}^{a^A_r+1} \max_{r'\in Q^A\cap[r-1]} \left\{\ln \frac{\alpha t'_a - \alpha t^A_{r'}}{a - a^A_{r'}}\right\} da \\
&\le |Q^A|\log\frac{2}{\alpha} + \frac{1}{\ln 2}\sum_{r\in Q^A} \int_{a=a^A_r}^{a^A_r+1} \max_{b<a} \left\{\ln \frac{\alpha t'_a - \alpha t'_b}{a - b}\right\} da \\
&= |Q^A|\log\frac{2}{\alpha} + \frac{1}{\ln 2}\int_{a=1}^{|Q^A|+1} \max_{b<a} \left\{\ln \frac{\alpha t'_a - \alpha t'_b}{a - b}\right\} da \\
&\le |Q^A|\log\frac{2}{\alpha} + \frac{1}{\ln 2} S_{\alpha T'}(|Q^A|+1) \\
&\le |Q^A|\log\frac{2}{\alpha} + \frac{1}{\ln 2}\alpha t^A \,.
\end{aligned}
$$

The first equation follows from the definition of u_r. The second equation follows from simple arithmetics (and the fact that $a^A_r - a^A_{r'} \ge 1$). The third equation follows from simple arithmetics, and in particular, from the fact that for every $a \in [a^A_r, a^A_r + 1]$ it holds that

$$\frac{t'_a - t^A_{r'}}{a - a^A_{r'}} \ge \frac{t^A_r - t^A_{r'}}{a^A_r + 1 - a^A_{r'}}.$$

The forth equation follows from trivial arithmetics (multiplying and dividing by α). The fifth equation follows by taking the component $\frac{2}{\alpha}$ outside of the summation. The sixth equation follows by replacing the log function with the ln function. The seventh equation follows from the fact that moving the max inside the integral can only increase the expression. The eighth equation follows from the fact that for every $r' \in Q^A \cap [r-1]$ it holds that $b \triangleq a^A_{r'} < a$. The ninth equation follows from basic properties or the integral. The tenth equation follows from the definition of $S_{\alpha T'}(|Q^A|+1)$. The final equation follows from Claim 48.

Similarly, one can argue that

$$\sum_{r\in Q^B} u_r \le |Q^B|\log\frac{2}{\alpha} + \frac{1}{\ln 2}\alpha t^B.$$

Thus,

$$\sum_{r\in Q} u_r \leq |Q| \log \frac{2}{\alpha} + \frac{1}{\ln 2}\alpha \mathrm{CC}(\Pi^{\mathcal{H}}_{\mathcal{A}}) \ .$$

Recall that the adversary $\mathcal{D}$ for the protocol Π sends the exact same messages as $\mathcal{A}$ does, excluding the hash values.

By definition of $\mathcal{A}$ and $\mathcal{D}$, for any round $r \notin Q$, the length of the r'th round message is the same for both $\Pi^{\mathcal{H}}_{\mathcal{A}}$ and $\Pi_{\mathcal{D}}$. For rounds $r \in Q$ we have that the messages in $\Pi^{\mathcal{H}}_{\mathcal{A}}$ and $\Pi_{\mathcal{D}}$ differ by 5 hash values. If there were no insertion or deletion errors on these hash values, then each of them would have been of size $2Cw_r + w_r + (w_r + 1) = (2C + 2)w_r + 1$. Thus,

$$\begin{aligned}
& \mathrm{CC}(\Pi^{\mathcal{H}}_{\mathcal{A}}) - \mathrm{CC}(\Pi_{\mathcal{D}}) - e \\
\leq & \sum_{r\in Q}(10C+10)w_r + 5 \\
= & \sum_{r\in Q}(10C+10)\left(\lceil \alpha\ell_r \rceil + \lceil u_r \rceil + 9\lceil \log\frac{1}{\gamma}\rceil + 6\right) + 5 \\
\leq & \sum_{r\in Q}(10C+10)\left(\alpha\ell_r + u_r + 9\log\frac{1}{\gamma} + 17\right) + 5 \\
= & (10C+10)\alpha\sum_{r\in Q}\ell_r + (10C+10)\sum_{r\in Q}u_r + \sum_{r\in Q}\left((90C+90)\log\frac{1}{\gamma} + (170C+175)\right) \\
\leq & (10C+10)\alpha\mathrm{CC}(\Pi) + (10C+10)\left(|Q|\log\frac{2}{\alpha} + \frac{1}{\ln 2}\alpha\mathrm{CC}(\Pi)\right) + \left((90C+90)\log\frac{1}{\gamma} + (170C+175)\right)|Q| \\
= & \left((10C+10) + \frac{(10C+10)}{\ln 2}\right)\alpha CC(\Pi) + \left((10C+10)\log\frac{1}{\alpha} + (90C+90)\log\frac{1}{\gamma} + (180C+185)\right)|Q| \\
\leq & 50C\alpha \cdot CC(\Pi) + 600C\log\frac{1}{\gamma}\cdot|Q| \ .
\end{aligned}$$

□

C.3 Proof of Claim 48

Fix any $R \in \mathbb{N}$ and any series $T = \{t_a\}_{a=1}^{R}$. For each $a \in \mathbb{R}^+$ let $P(a)$ be the set of points $b < a$ that maximize $\frac{t_a - t_b}{a-b}$.

Claim 49. *For any $r \in \mathbb{Z}$ and $\eta, \eta' \in (0,1)$, it holds that $r+\eta \in P(a)$ if and only if $r+\eta' \in P(a)$.*

Proof. First consider the case where $r = a - 1$. In this case,

$$\frac{t_a - t_{a-1+\eta}}{1-\eta} = \frac{t_a - (1-\eta)t_{a-1} - \eta t_a}{1-\eta} = t_a - t_{a-1} \ .$$

Since this expression does not depend on η, we conclude that $r + \eta \in P(a)$ if and only if $r + \eta' \in P(a)$.

Now consider $r < a - 1$. In this case, basic arithmetics shows that

$$\begin{aligned}
\frac{t_a - t_{r+\eta}}{a-r-\eta} &= \frac{t_a - (1-\eta)t_r - \eta t_{r+1}}{a-r-\eta} \\
&= \frac{\eta(a-r-1)}{a-r-\eta}\cdot\frac{t_a - t_{r+1}}{a-r-1} + \left(1 - \frac{\eta(a-r-1)}{a-r-\eta}\right)\frac{t_a - t_r}{a-r}
\end{aligned}$$

Therefore, $\frac{t_a - t_{r+\eta}}{a-r-\eta}$ is a non trivial convex combination of $\frac{t_a - t_{r+1}}{a-r-1}$ and $\frac{t_a - t_r}{a-r}$. Thus, if $r \in P(a)$ and $r+1 \in P(a)$, then for any η we get that $r+\eta \in P(a)$. On the other hand if r or $r+1$ are not in $P(a)$, then it must be the case that $r+\eta \notin P(a)$ for every η, since any (non-trivial) convex combination of any two elements is smaller than the maximum of these elements. □

Claim 50. $\forall a \in \mathbb{R}^+, \forall b \in P(a), S_T(a) \leq S_T(b) + (a-b)\ln\frac{e(t_a - t_b)}{a-b}$.

Before proving Claim 50, we prove that this claim implies Claim 48. We need to prove that $\forall a \in \mathbb{N}$,

$$S_T(a) \leq e \cdot t_a$$

To this end, consider the following recursive process. Let $a_0 = a$ and $a_{i+1} = \min\{P(a_i)\}$. By Claim 49 we have that a_{i+1} is an integer, and since by definition $\min\{P(a_i)\}) < a_i$, we conclude that a_{i+1} is in $\{0, 1, \ldots, a_i - 1\}$. Thus the process must end after $k \leq a$ steps with $a_k = 0$. Therefore, by a recursive application of Claim 50,

$$\begin{aligned} S_T(a) &= S_T(a_0) \leq S_T(0) + \sum_{i=0}^{k}(a_i - a_{i+1})\ln\frac{e(t_{a_i} - t_{a_{i+1}})}{a_i - a_{i+1}} \\ &\leq 0 + \sum_{i=0}^{k}(a_i - a_{i+1})\frac{e(t_{a_i} - t_{a_{i+1}})}{a_i - a_{i+1}} \\ &= \sum_{i=0}^{k} e(t_{a_i} - t_{a_{i+1}}) \\ &= e(t_{a_0} - t_{a_k}) = e \cdot t_a, \end{aligned}$$

as desired.

It thus remains to prove Claim 50.

Proof of Claim 50. Fix $a \in \mathbb{R}^+$ and fix $b \in P(a)$. We say that a point $c \in [0, a]$ is interesting if there exists $d \in \mathbb{R}$ such that c is the largest element satisfying $d \in P(c)$. Claim 49 implies that there are at most $2a$ interesting points.

The proof is by induction on the number of interesting points. The base case, when there are no interesting points, holds since in this case $a = 0$ and $S(a) = 0$.

Consider the case where $b = \max P(a)$, and let c be the interesting point before a.

We now show that $b \in P(r)$ for all $r \in [a, c)$. To this end, Fix such r and let $d \in P(r)$. We have that $\max P^{-1}(d)$ is an interesting point that is at least r which can be only a. Since $b = \max P(a)$ we have that $d \leq b$. Since $r \notin P(a)$ we have that $\frac{t_a - t_r}{a-r} < \frac{t_a - t_b}{a-b}$. We can write $\frac{t_a - t_b}{a-b}$ as a weighted average of $\frac{t_a - t_r}{a-r}$ and $\frac{t_r - t_b}{r-b}$ as follows

$$\frac{t_a - t_b}{a-b} = \frac{a-r}{a-b}\frac{t_a - t_r}{a-r} + \frac{r-b}{a-b}\frac{t_r - t_b}{r-b}\ .$$

Since this average is greater than $\frac{t_a - t_r}{a-r}$, we have that $\frac{t_r - t_b}{r-b}$ is the larger elements in the above average. Assume towards contradiction that $b \notin P(r)$. Thus $\frac{t_r - t_d}{r-d} > \frac{t_r - t_b}{r-b}$ and so

$$\begin{aligned} \frac{t_a - t_d}{a-d} &= \frac{a-r}{a-d}\cdot\frac{t_a - t_r}{a-r} + \frac{r-d}{a-d}\cdot\frac{t_r - t_d}{r-d} \\ &> \frac{a-r}{a-d}\cdot\frac{t_a - t_r}{a-r} + \frac{r-d}{a-d}\cdot\frac{t_r - t_b}{r-b} \\ &\geq \frac{a-r}{a-b}\cdot\frac{t_a - t_r}{a-r} + \frac{r-b}{a-b}\cdot\frac{t_r - t_b}{r-b} \\ &= \frac{t_a - t_b}{a-b}\ , \end{aligned}$$

When the last inequality follows from increasing the weight of the larger element in a weighted sum. This contradict the fact that $b \in P(a)$.

We got that for every d and $r > c$ we have that $f(r) = \frac{t_r - t_b}{r-b} - \frac{t_r - t_d}{r-d} \geq 0$. From the continuity of f around c, we have that $f(c) \geq 0$, and so $b \in P(c)$. Thus, by induction hypothesis $S_T(c) \leq S_T(b) + (c-b)\ln\frac{t_c - t_b}{c-b}$. By simple calculations, and using the fact that $\int \ln\frac{A}{x}dx = x\ln\frac{eA}{x}$,

$$\begin{aligned} S_T(a) &\leq S_T(c) + \int_c^a \ln\frac{t_r - t_b}{r-b}dr \\ &\leq S_T(b) + (c-b)\ln\frac{e(t_a - t_b)}{c-b} + \int_c^a \ln\frac{t_a - t_b}{r-b}dr \\ &= S_T(b) + (c-b)\ln\frac{e(t_a - t_b)}{c-b} + (a-b)\ln\frac{e(t_a - t_b)}{a-b} - (c-b)\ln\frac{e(t_a - t_b)}{c-b} \\ &= S_T(b) + (a-b)\ln\frac{e(t_a - t_b)}{a-b} \end{aligned}$$

Now consider the case that $\max P(a) = c \neq b$. By definition $\frac{t_a - t_b}{a-b} = \frac{t_a - t_c}{a-c}$ and so $\frac{t_a - t_c}{a-c} = \frac{t_c - t_b}{c-b}$. First we observe that $b \in P(c)$. This is the case, since otherwise, there was d such that $\frac{t_c - t_d}{c-d} > \frac{t_c - t_b}{c-b}$, and so $\frac{t_a - t_d}{a-d} > \frac{t_a - t_b}{a-b}$, contradicting the fact that $b \in P(a)$. Hence, by the induction hypothesis

$$S_T(c) \leq S_T(b) + (c-b)\ln\frac{e(t_c - t_b)}{c-b}.$$

By the previous case we saw that

$$S_T(a) \leq S_T(c) + (a-c)\ln\frac{e(t_a - t_c)}{a-c},$$

and we get

$$\begin{aligned} S_T(a) &\leq S_T(c) + (a-c)\ln\frac{e(t_a - t_c)}{a-c} \\ &\leq S_T(b) + (c-b)\ln\frac{e(t_c - t_b)}{c-b} + (a-c)\ln\frac{e(t_a - t_c)}{a-c} \\ &= S_T(b) + (a-b)\ln\frac{e(t_a - t_b)}{a-b} \end{aligned}$$

□

D Proof of Theorem 17

Proof. The fact that the number of bits sent in Π' is at least $t_{\min}$ follows from the definition.

We note that a priori it may seem that the parties may not agree on which of the messages are system messages. However, in the following claim, we show that this is not the case.

Claim 51. *Throughout the protocol the parties agree on k and $\{s_i, P_i, r_i\}_{i=1}^k$.*

Proof. Assume without lost of generality that Alice received the last message of the protocol, and let K be the last value of the variable k of Alice. We first show that both parties agree on the values of $\{s_i, P_i, r_i\}_{i=1}^K$.

Let $m_1^A, \ldots, m_K^A$ be the system messages sent by Alice (including echoes), and let $m_1^B, \ldots, m_{K'}^B$ be the system messages sent by Bob (including echoes). The fact that m_K^A cannot be corrupted, follows from the following claim.

Claim 52. $|m_K^A| > 12\epsilon d \log d \mathrm{CC}(\Pi')$.

We defer the proof of Claim 52 for later.

Now consider the case that when Bob receives the message m_K^A from Alice, his variable k satisfies $|m_K^A| \geq b^k W$. In this case Bob will parse m_K^A as a system message. Since by our assumption, Bob does not halt, then both parties agree on $\{s_i, P_i, r_i\}_{i=1}^K$.

Now consider the case where $|m_K^A| < b^k W$. In this case, we have that $k > K$, and thus the message m_K^B was sent before m_K^A. Since $|m_K^B| = |m_K^A| \geq \epsilon \mathrm{CC}(\Pi')$, we have also that m_K^B hasn't been corrupted. Thus, Alice will parse m_K^B as a system message, and since she does not halt, we have that both parties agree on $\{s_i, P_i, r_i\}_{i=1}^K$.

We next argue that Bob does not send the message m_{K+1}^B. We assume towards contradiction that Bob does send this message. Since $|m_{K+1}^B| \geq |m_K^A|$ this message hasn't been corrupted, and hence Alice parses it as a system message. Since by our assumption, Alice does not respond with a system message, we conclude that there were more than $\frac{1}{bd \log d}$ fraction of errors in the first $\frac{1}{2} b^K W$ bits of the protocol. Observe that $|m_K^A| \leq 6b^{K-1}W$, which follows from the fact that

$$|m_K^A| = |(s_1, \ldots, s_K, P_1, \ldots, P_K, r_1, \ldots, r_K, 1)| = 1 + \sum_{j=0}^{K-1} 3b^j W \leq 6b^{K-1}W.$$

Thus, the total number of corruptions is at least,

$$\frac{\frac{1}{2} b^K W}{bd \log d} \geq \frac{1}{12 d \log d} |m_K^A| > \epsilon \mathrm{CC}(\Pi') \ ,$$

in contradiction.

We conclude that both parties send K system messages. Since the parties agree on $r_1, \ldots, r_K$, they agree when the system messages were sent. In particular for each $i < K$, they both send the same i'th system message in consecutive rounds. Thus, throughout the protocol the parties always agree on k and $\{s_i, P_i, r_i\}_{i=1}^k$.

Proof of Claim 52. First we note that the total communication in Alice's view, denoted by $\mathrm{CC}^A(\Pi'_{\mathcal{A}})$, is at least $(1-\epsilon)\mathrm{CC}(\Pi'_{\mathcal{A}})$, which in turn is at least $\frac{1}{2}\mathrm{CC}(\Pi'_{\mathcal{A}})$. Since each system message that Alice received is either an echo of a system message she sent, or was echoed by her, the total communication of system messages according to Alice, is at most $2\sum_{i=1}^K |m_i^A|$. All the non-system messages she stores in T^A, possibly without their last bit. Thus, the total volume of non-system messages according to Alice is at most $2|T^A|$. Finally, since Alice does not send any more system messages after m_K^A, we have that $|T^A| \leq \frac{b^K W}{400 C \alpha}$. Putting it all together, we

get,

$$\begin{aligned}
\mathrm{CC}(\Pi'_{\mathcal{A}}) &\leq 2\mathrm{CC}^A(\Pi'_{\mathcal{A}}) \\
&\leq 4|T^A| + 4\sum_{i=1}^{K} |m_K^A| \\
&\leq 4 \cdot \frac{b^K W}{400C\alpha} + 4\sum_{i=0}^{K-1}\left(1 + \sum_{j=0}^{i} 3b^j W\right) \\
&\leq \frac{4b}{\alpha} \cdot b^{K-1} W + 24\sum_{i=0}^{K-1} b^i W \\
&\leq \left(\frac{4b}{3\alpha} + 8\right)\sum_{i=0}^{K-1} 3b^i W \\
&\leq \left(\frac{4b}{3\alpha} + 8\right)|m_K^A| \\
&< \frac{|m_K^A|}{12\epsilon d \log d}\,,
\end{aligned}$$

where the last inequality follows from the fact that $\epsilon \leq \frac{b}{10\alpha d \log d}$, $b > 2$ and $\alpha < \frac{1}{3200}$. □

□

Simulation. We next define an adversary $\mathcal{D}$ for the protocol Π that satisfies the requirements of Theorem 17.

To this end, we first define an adversary $\mathcal{A}'$ for the protocol $\Pi^{\mathcal{H}}$ that emulates the adversary $\mathcal{A}$ in Π'. The adversary $\mathcal{A}'$ emulates (in his head) a transcript corresponding to Π', by emulating the system messages (using the shared random string) and the bits added to long messages (as in Π'), and applies $\mathcal{A}$ to these messages. The corruption strategy of $\mathcal{A}'$ is the induced corruption to the messages corresponding to $\Pi^{\mathcal{H}}$.

The fact that the adversary $\mathcal{A}'$ is well defined follows from Claim 51, which guarantees that the parties in $\Pi'_{\mathcal{A}}$ always agree on the string used as shared randomness, and on the rounds where system messages are sent.

As defined in Section 5.4, we let $\mathcal{D}$ be the adversary for the protocol Π, that sends the exact same messages as $\mathcal{A}$, excluding the hash values.

Note that by definition, when $\Pi'_{\mathcal{A}}$ ends, both Alice and Bob (separately) can efficiently compute their view of the transcript $\Pi_{\mathcal{D}}$. Moreover, the total volume and number of messages corrupted by $\mathcal{D}$ are bounded by those of $\mathcal{A}'$, which in turn are bounded by the volume and number of messages corrupted by $\mathcal{A}$.

Communication Complexity. We next bound the communication complexity of $\Pi'_{\mathcal{A}}$. We will have 3 budgets: $\mathcal{O}$ for the original messages of $\Pi^{\mathcal{H}}$, $\mathcal{B}$ for the extra bit added to messages of length $\geq b^k W$, and $\mathcal{S}$ for system messages. Thus,

$$\mathrm{CC}(\Pi'_{\mathcal{A}}) = \mathcal{O} + \mathcal{B} + \mathcal{S}.$$

Note that by definition $\mathcal{B} \leq \frac{\mathcal{O}+\mathcal{S}}{W}$.

We next argue that it suffices to prove that

$$\mathcal{O} + \mathcal{S} \leq (1 + 24\alpha) \cdot \mathrm{CC}(\Pi^{\mathcal{H}}_{\mathcal{A}'}).$$

This follows from Lemma 16 and the assumption that $W = \alpha t_{\min} \geq \frac{1}{\alpha}$, as follows:

$$\begin{aligned}
\mathrm{CC}(\Pi'_{\mathcal{A}}) &= \mathcal{O} + \mathcal{B} + \mathcal{S} \\
&\leq (1 + \frac{1}{W})(1 + 24\alpha) \cdot \mathrm{CC}(\Pi^{\mathcal{H}}_{\mathcal{A}'}) \\
&\leq (1+\alpha)(1+24\alpha)\left((1+50C\alpha)CC(\Pi_{\mathcal{D}}) + \left(\epsilon + \frac{600C\log\frac{1}{\gamma}}{d/2}\right)CC(\Pi'_{\mathcal{A}})\right) \\
&\leq (1+80C\alpha)\,CC(\Pi_{\mathcal{D}}) + 1201C\alpha \cdot \mathrm{CC}(\Pi'_{\mathcal{A}})\,.
\end{aligned}$$

Thus we got that

$$\mathrm{CC}(\Pi'_{\mathcal{A}}) \leq \frac{1+60C\alpha}{1-1201C\alpha} \cdot CC(\Pi_{\mathcal{D}}) \leq (1+2600C\alpha) \cdot CC(\Pi_{\mathcal{D}})$$

as desired.

Note that $\mathcal{O} = \mathrm{CC}(\Pi^{\mathcal{H}}_{\mathcal{A}'})$. Thus, it remains to prove that

$$\mathcal{S} \leq 24\alpha \cdot \mathrm{CC}(\Pi^{\mathcal{H}}_{\mathcal{A}'}) = 24\alpha \cdot \mathcal{O}.$$

Denote by

$$m_1, m'_1, m_2, m'_2, \ldots, m_K, m'_K$$

the system messages in Π', where for every $i \in [K]$, the message m'_i is an echo of m_i, and thus these messages are identical (follows from Claim 51).

Note that for every $i \in [K-1]$, it holds that

$$|m_{i+1}| \geq |m_1| + \ldots + |m_i|,$$

and in particular,

$$|m_1| + \ldots + |m_{K-1}| + |m_K| \leq 2|m_K|.$$

Moreover, note that when m_K is sent (in step 4(e)ii), it holds that

$$|m_K| \leq 6b^{K-1}W \leq 6\alpha\mathcal{O}' \leq 6\alpha\mathcal{O},$$

where $\mathcal{O}'$ is the local transcript corresponding to $\Pi^{\mathcal{H}}$ after the party sending m_K will send its next message in Π. Thus, we conclude that overall

$$\mathcal{S} = 2|m_1| + \ldots + 2|m_K| \leq 4|m_K| \leq 24\alpha \cdot \mathcal{O},$$

as desired.

Round Complexity. We denote by K the number of system messages that were sent in Π' by each of the players. Note that

$$R(\Pi'_{\mathcal{A}}) = R(\Pi^{\mathcal{H}}_{\mathcal{A}'}) + 2K = R(\Pi_{\mathcal{D}}) + 2K.$$

Thus, it suffices to prove that

$$K \leq \log_b \mathrm{CC}(\Pi_{\mathcal{D}}).$$

Since the parties send the K'th system message, we have that $\mathrm{CC}(\Pi^{\mathcal{H}}_{\mathcal{A}'}) \geq \frac{b^{K-1}W}{800C\alpha}$. This, together with our previous bound on the communication complexity of Π', and together with our assumption that $\alpha \leq \frac{1}{3200C}$ implies that

$$\begin{aligned}
&K \leq \log_b \left(800C\alpha \mathrm{CC}(\Pi^{\mathcal{H}_{\mathcal{A}'}})\right) + 1 \leq \\
&\log_b \left(1600C\alpha \mathrm{CC}(\Pi^{\mathcal{H}}_{\mathcal{A}'})\right) \leq \\
&\log_b \left(1600C\alpha \mathrm{CC}(\Pi'_{\mathcal{A}})\right) \leq \\
&\log_b \left(1600C\alpha(1 + 2600C\alpha)\mathrm{CC}(\Pi_{\mathcal{D}})\right) \leq \\
&\log_b \mathrm{CC}(\Pi_{\mathcal{D}}) \,,
\end{aligned}$$

as desired.

Bounding hash collisions. It remains to prove Item 5 in Theorem 17. To this end, we finally define

$$x = \{x_{i,r}\}_{i\in[5],r\in\mathbb{N}} \,,$$

where each $x_{i,r}$ is defined as a function of $(s_1, \ldots, s_k, r_1, \ldots, r_k)$ that were sent in Π' up until the point where round r of $\Pi^{\mathcal{H}}$ is simulated.

Fix a round r and let n be the maximal such that $r_n \leq r$. We partition s_n into 5 equal parts,[22]

$$s_n = (s_{n,1}, \ldots, s_{n,5}),$$

where for each $i \in [5]$, the string $s_{n,i}$ is used to generate $x_{r,i}$, the seed for the i'th hash function used in round r, as follows: We define

$$L \triangleq \left(\frac{|G(s_{n,i})|}{2}\right)^{1/3},$$

where G is the pseudo-random generator function from Lemma 13. Thus $|G(s_{n,i})| = 2L^3$. Partition

$$G(s_{n,i}) = (x_{n,i,1}, \ldots, x_{n,i,L})$$

where for each $r \in [L]$, it holds that $|x_{n,i,r}| = 2L^2$. We define $x_{i,r} \triangleq x_{n,i,r}$ for every $r \in [L]$. We remark that the protocol never uses $x_{i,r}$ for $r > L$, however we define $x_{i,r}$ for $r > L$ to be uniform only for the simplicity of the analysis (to avoid dealing with edge cases).

Recall, that in Protocol $\Pi^{\mathcal{H}}$, it suffices to use $x_{i,r}$ of length $2t_r^2$, where t_r is the total communication up to round r in $\Pi^{\mathcal{H}}$. This is the case since we used the hash function from Lemma 14, which takes as input a string $\{0,1\}^{\leq L}$ to a single output bit, using a seed of length $2L$. In round r of the protocol $\Pi^{\mathcal{H}}$, we applied this function at most t_r times on inputs of length at most t_r. Hence, in total we need a seed of length $2t_r \cdot t_r = 2 \cdot t_r^2$.

Thus we need to prove the following claim.

Claim 53. $L \geq t_r$.

Proof. Note that it is always the case that

$$|s_n| \geq W = \alpha t_{\min} \geq 250C \log d,$$

where the latter follows from the definition of $t_{\min}$ (see Equation (5)). This implies that

$$|s_{n,i}| \geq 50C \cdot \log d,$$

[22] We assume without loss of generality that 5 divides $|s|$.

which in turn implies that

$$L \triangleq \left(\frac{|G(s_{n,i})|}{2}\right)^{1/3} \geq \left(\frac{2^{50 \cdot \log d}}{2}\right)^{1/3} \geq d^2.$$

Thus, if $t_r < d^2$, then indeed $t_r \leq L$, as desired.

Therefore, assume that $t_r \geq d^2$. By the protocol,

$$|s_n| = b^{n-1}W \geq \frac{400C\alpha}{b} t_r \geq \frac{400C}{d} t_r \ ,$$

where the fact that $t_r \leq \frac{b^n W}{400C\alpha}$ follows from the definition of the protocol Π', and the latter inequality follows from the fact that $d \geq \frac{b}{\alpha}$ (see Equation (4)). Thus,

$$L \triangleq \left(\frac{|G(s_{n,i})|}{2}\right)^{1/3} \geq \left(\frac{2^{\frac{400t_r}{d}}}{2}\right)^{1/3} \geq \left(\frac{2^{400\sqrt{t_r}}}{2}\right)^{1/3} \geq t_r,$$

where the last inequality follows from a straightforward calculus, and the second to last inequality follows from the fact that $\frac{t_r}{d} \geq \sqrt{t_r}$, which in turn follows from our assumption that $t_r \geq d^2$.

□

Having defined the random string x and the adversary $\mathcal{A}'$, we are now ready to state the following lemma, which immediately implies Item 5 (as we explain after the lemma statement).

Lemma 54. *Consider the protocol $\Pi^{\mathcal{H}}_{\mathcal{A}'}$ with the random string x defined above. Then*

1. *With probability $\geq 1 - 20 \cdot 2^{-\frac{\gamma}{3d}t}$,*

$$|E \setminus E_0| \leq 35\gamma t.$$

2. *With probability $\geq 1 - 80r \cdot 2^{-\frac{7\gamma^8}{d}r}$,*

$$|E \setminus E_0|' \leq 100\gamma r.$$

Recall that $E \setminus E_0$ is the set of all messages with a hash collision.

The reason this lemma implies Item 5 follows from the fact that the volume of messages with hash collisions in $\Pi_{\mathcal{D}}$ is at most the volume of messages with hash collisions in $\Pi^{\mathcal{H}}_{\mathcal{A}'}$, and the communication complexity in $\Pi^{\mathcal{H}}_{\mathcal{A}'}$ is at most twice the communication complexity in $\Pi_{\mathcal{D}}$ (follows from Item 2 together with the bound on α). Similarly the number of rounds with hash collisions in $\Pi_{\mathcal{D}}$ is at most the number of rounds with hash collisions in $\Pi^{\mathcal{H}}_{\mathcal{A}'}$.

In order to prove this Lemma, we need to prove that the random string x and the adversary $\mathcal{A}'$ have the following properties.

Claim 55. *For every $i \in [5]$, every $k \in \mathbb{N}$, and every $r_1, \ldots, r_k \in \mathbb{N}$, and every $r_0 \in \mathbb{N}$, the distribution $x(s_1, \ldots, s_k, r_1, \ldots, r_k)$, where $\{s_i\}_{i=1}^k$ are uniform, has the property that $\{x_{i,r}\}_{r \geq r_0}$ is $2^{-\frac{80t_{r_0}}{d}}$-biased.*

Claim 56. *The adversary $\mathcal{A}'$ (defined above) for the protocol $\Pi^{\mathcal{H}}$ satisfies that for any $t \in [\frac{W}{2}, CC(\Pi^{\mathcal{H}}_{\mathcal{A}'})]$, the volume of corrupted messages in the first t bits of $\Pi^{\mathcal{H}}_{\mathcal{A}'}$ is at most $\frac{t}{d \log d}$.*

proof of Claim 55. Let n be the maximal that satisfies $r_n \leq r_0$. By our construction the variables can be partitioned into disjoint sets

$$\{x_{i,r}\}_{r \geq r_0} = \{x_{i,r}\}_{r=r_0}^{r_{n+1}} \cup \{x_{i,r}\}_{r=r_{n+1}+1}^{r_{n+2}} \cup \cdots \cup \{x_{i,r}\}_{r \geq r_k+1}$$

It suffices to prove that for every $j \in \{n, \ldots, k\}$, the joint distribution of $\{x_{i,r}\}_{r=r_j+1}^{r_{j+1}}$ is $2^{-\frac{t_{r_0}}{40d}}$-biased (since it is independent of all the rest).

According to the protocol, t_{r_0} is at most $\frac{b^n W}{400C\alpha}$, and recall that $d \geq b/\alpha$. Thus, for any $j \geq n$ we have

$$|s_j| = b^{j-1}W \geq b^{n-1}W = \frac{400C\alpha}{b} \cdot \frac{b^n W}{400C\alpha} \geq \frac{400C\alpha}{b} t_{r_0} \geq \frac{400C}{d} t_{r_0} \ .$$

Hence, the seed $s_{j,i}$ is of length at least $\frac{80Ct_{r_0}}{d}$. Since s_j, and hence $s_{j,i}$, is independent of all the other seeds $\{s_i\}_{i \neq j}$, then even conditioned on any fixing of the latter, by Lemma 13, the set $\{x_{i,r}\}_{r=r_j+1}^{r_{j+1}}$ is $2^{-\frac{80t_{r_0}}{d}}$-biased, as desired.

□

Proof of Claim 56. Fix some $t \in [\frac{W}{2}, CC(\Pi^{\mathcal{H}})]$. First consider the case where $t \geq \frac{1}{2}b^{K-1}W$, where K is the final value of k in the protocol. Since there are at most $\epsilon \mathrm{CC}(\Pi'_{\mathcal{A}})$ errors on the first t bits, we have that the fraction of errors is at most

$$\frac{\epsilon \mathrm{CC}(\Pi'_{\mathcal{A}})}{t} \leq \frac{\epsilon(1 + 2600C\alpha)\mathrm{CC}(\Pi_{\mathcal{D}})}{t} \leq \frac{2\epsilon \mathrm{CC}(\Pi^{\mathcal{H}}_{\mathcal{A}})}{t} \leq \frac{2\epsilon \frac{b^K W}{400C\alpha}}{\frac{1}{2}b^{K-1}W} = \frac{\epsilon b}{100C\alpha} \leq \frac{1}{d \log d} \ ,$$

where the first inequality follows from our bound on the communication complexity, the second inequality follows from the fact that $2600C\alpha \leq 1$ and $\mathrm{CC}(\Pi_{\mathcal{D}}) \leq \mathrm{CC}(\Pi^{\mathcal{H}}_{\mathcal{A}})$, and the last inequality follows from the bound $\epsilon \leq \frac{\alpha}{bd \log d}$.

Now consider the case that $t \leq \frac{1}{2}b^{K-1}W$ and let k be such that $\frac{1}{2}b^{k-1}W \leq t \leq \frac{1}{2}b^k W$. By Claim 51, the parties do not halt immediately after the $(k+1)$-st system message. Therefore, on the first $\frac{1}{2}b^k W$ bits of the protocol there were less than $\frac{1}{2}b^k W \cdot \frac{1}{bd \log d}$ errors. Thus, the fraction of errors on the first t bits is at most

$$\frac{\frac{1}{2}b^k W \cdot \frac{1}{bd \log d}}{\frac{1}{2}b^{k-1}W} = \frac{1}{d \log d} \ .$$

□

We are now ready to prove Lemma 54. The proof follows by an adaptation of the proof of Lemma 15, presented in Appendix C.1. In what follows we use the notations and definitions from Appendix C.1. Recall that the proof of Lemma 54, follows by proving four claims: Claim 42, Claim 44, Claim 45, and Claim 46.

Note that Claims 44 and 45 still hold with respect to our new hash functions, since these claims do not depend on the public randomness. Thus, to prove Lemma 54, it suffice to prove an adapted version of Claims 42 and 46, stated below.

Claim 57. $\forall k \in [5]$, $\Pr[|E_k| \leq 5\gamma t] \geq 1 - 4 \cdot 2^{-\frac{7\gamma}{d}t}$.

Claim 58. $\forall k \in [5]$, $\Pr\left[|E_k|' \geq 16\gamma r\right] \leq 8r \cdot 2^{-\frac{7\gamma^8}{d}r}$.

We now fix k. Consider a random fixing of the shared randomness x and the private randomness r_A and r_B (corresponding to Alice and Bob, respectively). This determines a transcript T for $\Pi^{\mathcal{H}}_{\mathcal{A}}$ (with x, r_A, r_B). In any non-corrupted round r with hash in T, we consider the vector $f^{2^{w_r}}_{x_{k,r}}(V_k)$ (where V_k is determined by x, r_A, r_B), and divide it into hash chunks of size $\log \frac{1}{\gamma}$.

We define $S = S_{g,x,r_A,r_B}$ to be the set of hash chunks, where for every non-corrupted message of length ℓ with hash, sent in the r'th round of the protocol (determined by (x, r_A, r_B)), S contains the first $g(\ell)$ hash chunks corresponding to $f^{2^{w_r}}_{x_{k,r}}(V_k)$ (where V_k corresponds to (x, r_A, r_B)), for some g such that $g(\ell) \leq 2^{w_r}/\log \frac{1}{\gamma}$. Let $S[t', t] = S_{g,x,r_A,r_B}[t', t]$ contain only the hash chunks in S that were sent after at least t' bits were sent, and at most t bits were sent.

The proofs of Claims 57 and 58 follow from the next technical claim.

Claim 59. *Let S, t, t' be as above, and let N be an upper bound on the number of hash chunks in S. Then, the probability that there are more than $2\gamma N$ chunks with hash collision in $S[t', t]$, is at most $2^{\frac{13}{d}t}\left(e^{-\frac{1}{3}\gamma N} + 2^{-\frac{40t'}{d}}\right)$.*

Proof. Fix the private randomness r_A and r_B of Alice and Bob, respectively. Given 5 subsets (of rounds with hash) $J_1, \ldots, J_5 \subseteq [\frac{2t}{d}]$, we denote $J = \{J_1, \ldots, J_5\}$, and define $\Pi^{\mathcal{H},J}$ to be the protocol that acts like $\Pi^{\mathcal{H}}$ with the following changes: For the r-th round with a hash, the protocol $\Pi^{\mathcal{H},J}$ acts like there is a hash collision on the variable V_j if and only if $r \in J_j$. Thus, in a sense, the protocol $\Pi^{\mathcal{H},J}$ does not depend on the public randomness x.

We next bound the number of hash chunks in $S[t', t]$, corresponding to the transcript $\Pi^{\mathcal{H},J}$, that have collisions with respect to the public randomness x. We emphasize that the protocol $\Pi^{\mathcal{H},J}$ is independent of whether or not there is a hash collision with respect to x, yet in $S[t', t]$ we count the number of hash collisions with respect to x.

Let $(\Pi^J)'$ denote the protocol obtained by simulating $\Pi^{\mathcal{H},J}$, as was defined in Section 6. Namely, $(\Pi^J)'$ is defined as Π' was, however rather than defining it with respect to $\Pi^{\mathcal{H}}$ it is defined with respect to $\Pi^{\mathcal{H},J}$. Note that the rounds $r_1, \ldots, r_k$ that are used to generate x in $(\Pi^J)'$ are *independent* of the random seeds $s_1, \ldots, s_k$. This is the case since the behavior of the protocol $\Pi^{\mathcal{H},J}$ does not depend on the random string x.

Therefore, for any oblivious adversary O that make at most $\frac{t}{d \log d}$ errors on the first t bits, when the random string x is uniform, by the Chernoff bound (see Lemma 11), the probability that there are more than $2\gamma N$ hash collisions (with respect to x) in S in the protocol $\Pi^{\mathcal{H},J}_O$, is at most $e^{-\frac{1}{3}\gamma N}$. By Lemma 14, when x comes from a $2^{-\frac{40t'}{d}}$-bias distribution (as opposed to uniform), this probability is at most $e^{-\frac{1}{3}\gamma N} + 2^{-\frac{40t'}{d}}$.

By Claim 43 there are at most $2^{\frac{3t}{d}}$ oblivious adversaries that make at most $\frac{t}{d \log d}$ errors in the first t bits. By the union bound over all such adversaries and over all $2^{\frac{10t}{d}}$ possible sets J, we have that for any such adversary and any set J, the probability that there are more than $2\gamma N$ hash chunks with hash collisions in $\Pi^{\mathcal{H},J}$ is at most

$$2^{\frac{3}{d}t + \frac{10}{d}t}(e^{-\frac{1}{3}\gamma N} + 2^{-\frac{40t'}{d}}) \leq 2^{\frac{13}{d}t}(e^{-\frac{1}{3}\gamma N} + 2^{-\frac{40t'}{d}}) \,.$$

We show that this bound holds also for $\Pi^{\mathcal{H}}_{\mathcal{A}'}$. Let O describe the messages that were corrupted due to the adversary $\mathcal{A}'$, and let J describe the rounds with hash collisions. Note that by Claim 56, O corrupts at most $\frac{t}{d \log d}$ bits from the first t bits of the transcript. By definition, $\Pi^{\mathcal{H}}_{\mathcal{A}'}$ acts exactly like $\Pi^{\mathcal{H},J}_O$.

Hence, we conclude that the number of hash chunks with hash collisions is at most $2\gamma N$, as required. □

We are now ready to prove Claim 57.

Proof of Claim 57. Let $S = S_{g,x,r_A,r_B}$ where $g(\ell) = \gamma^{-1}\ell$. Namely, S contains the first $\gamma^{-1}\ell$ hash chunks of each message of length ℓ that was sent with a hash. This is well defined since $2^{w_r} > \gamma^{-1} \log \frac{1}{\gamma}\ell$.

Let $S_i = S[\frac{t}{2^{i+1}}, \frac{t}{2^i}]$. We have that $|S_i| \leq \frac{t}{\gamma 2^{i+1}}$. By Claim 59 the probability that S_i has more than $\frac{t}{2^i}$ hash chunks with hash collisions is at most

$$\begin{aligned}
&2^{\frac{13t}{2^i d}} \left(e^{-\frac{t}{3 \cdot 2^{i+1}}} + 2^{-\frac{40t}{d2^{i+1}}} \right) \leq \\
&2^{\frac{13t}{2^i d}} \left(2 \cdot 2^{-\frac{40t}{d2^{i+1}}} \right) \leq \\
&2 \cdot 2^{\left(13 - \frac{40}{2}\right)\frac{t}{d2^i}} = 2 \cdot 2^{-\frac{7}{d2^i}t} \,.
\end{aligned}$$

Consider all the messages that were sent after $\frac{t}{2^{i+1}}$ bits were sent and before $\frac{t}{2^i}$ bits were sent, and consider the volume of all these messages for which all the hash chunks in S have a hash collision. Thus, with probability $\geq 1 - 2 \cdot 2^{-\frac{7}{2^i d}t}$, we have that this volume is at most $\frac{\gamma t}{2^i}$. By the union bound, we have that the probability that for any $0 \leq i \leq \log \frac{1}{\gamma}$ the above holds is at least,

$$1 - \sum_{i=1}^{\log \frac{1}{\gamma}} 2 \cdot 2^{-\frac{7}{2^i d}t} \geq 1 - 4 \cdot 2^{-\frac{7}{2^{\log \frac{1}{\gamma}} d}t} = 1 - 4 \cdot 2^{-\frac{7\gamma}{d}t} \,.$$

Moreover, in the first $\frac{t}{2^{\log \frac{1}{\gamma}}} = \gamma t$ bits of the protocol, the volume of E_k is clearly at most γt. Thus, with probability $\geq 1 - 4 \cdot 2^{-\frac{7\gamma}{d}t}$ the total volume of hash collisions is at most

$$\gamma t + \sum_{i=0}^{\log \frac{1}{\gamma}} \frac{\gamma t}{2^i} \leq 3\gamma t \,.$$

□

Proof of Claim 58. The proof of Claim 58 follow the footsteps of the proof of Claim 46. Recall that in the proof of Claim 46, we bound the number of hash collisions on messages received by Alice, and messages received by Bob, separately. For the former, we partition the messages received by Alice into regimes, and classify some of these regimes as *heavy* and the others as *light*. Loosely speaking, a heavy regime is one where the number bits that Alice received during this regime consists of a large fraction ($\frac{\gamma}{2}$) of the bits that Alice received so far. We showed that the total number of hashes in the light regimes is bounded by $2\gamma r$. The same is also true in our setting.

Thus, it remains to prove that with probability $4r \cdot 2^{-\frac{7\gamma^8 r}{d}}$, the number of rounds with hash collisions sent by Alice in all heavy regimes is at most $6\gamma r$.

To this end, for any heavy regime we let t denote the number of bits that Alice received in the protocol until the end of this regime. As in the proof of Claim 46, we distinguish between the case where $t \geq \gamma r$ and the case where $t < \gamma r$. As we proved, the number of messages with hash that Alice sends overall in all the regimes for which $t < \gamma r$, is at most γr.

Therefore, to prove that with probability $4r \cdot 2^{-\frac{7\gamma^8 r}{d}}$, the number of rounds with hash collisions sent by Alice in all heavy regimes is at most $6\gamma r$, it suffices to prove that with probability $4r \cdot 2^{-\frac{7\gamma^8 r}{d}}$, the number of rounds with hash collisions sent by Alice in all heavy regimes for which $t \geq \gamma r$ is at most $5\gamma r$. Namely, by the union bound, it suffices to prove the following claim (which is an analog of Claim 47).

Claim 60. *For any adversary and any heavy regime, the probability that the regime has more than $5\gamma\Delta a$ rounds of hash collisions is at most $4 \cdot 2^{-\frac{7\gamma^7 t}{d}}$, where t is the number of bits that Alice received in the protocol until the end of this regime.*

Proof. Fix an adversary and a heavy regime. Let Δt be the volume of messages in this regime (i.e., messages received by Alice in this regime), and let Δa be the number of messages that Alice sent with hashes in this regime. Let $U = \frac{\Delta t}{\Delta a}$. This definition of U is similar to Equation (34).

Let S be the set of the first $64\gamma^{-2}U$ hash chunks in each message that Alice sends with hash in this regime. There are enough bits of hash in each such message since,

$$2^{w_r} = 2^{\max_{r' \in Q \cap [r-1]} \log \frac{t_r - t_{r'}}{a_r - a_{r'}} + 9 \log \frac{1}{\gamma} + 6} = 64\gamma^{-9} \max_{r' \in Q \cap [r-1]} \frac{t_r - t_{r'}}{a_r - a_{r'}} \geq 64\gamma^{-9} U' \geq 64\gamma^{-3} U \ ,$$

where the latter is greater than $64\gamma^{-2}U \cdot \log \frac{1}{\gamma}$, as desired, where U' is defined in the proof of Claim 46, and where the latter inequality follow from Equation (35). Thus, the number of hash chunks in S is

$$|S| = \Delta a \cdot 64\gamma^{-2}U = 64\gamma^{-2}\Delta t \geq 32\gamma^{-1}t, \tag{37}$$

where the latter inequality follows from the definition of heavy regime, which asserts that $\Delta t > \frac{1}{2}\gamma t$.

Let $k = 7 \log \frac{1}{\gamma}$. For each $0 \leq i \leq k$, let S_i be the part of S that was sent by Alice, after receiving from Bob at least $\frac{t}{2^{i+1}}$ bits in this regime, and at most $\frac{t}{2^i}$ bits in the regime. Let S' be the remaining part of S. By Claim 59, we have that the probability that in each S_i, there are more than $2\gamma|S_i| + \frac{\gamma}{2^i}|S|$ hash chunks with hash collisions is at most $2^{\frac{13}{2^i d}t}\left(e^{-\frac{\gamma}{3 \cdot 2^i}|S|} + 2^{-\frac{40t}{2^{i+1}d}}\right)$. By union bound the probability that for all i, the number of hash collisions in S_i is at most $2\gamma|S_i| + \frac{\gamma}{2^i}|S|$ is,

$$\begin{aligned}
\sum_{i=0}^{k} 2^{\frac{13}{2^i d}t}\left(e^{-\frac{\gamma}{3 \cdot 2^i}|S|} + 2^{-\frac{40t}{2^{i+1}d}}\right) &\leq \sum_{i=0}^{k} 2^{\frac{13}{2^i d}t}\left(e^{-\frac{32}{3 \cdot 2^i}t} + 2^{-\frac{20t}{2^i d}}\right) \\
&\leq 2 \cdot \sum_{i=0}^{k} 2^{\frac{13}{2^i d}t} \cdot 2^{-\frac{20t}{2^i d}} \\
&= 2\sum_{i=0}^{k} 2^{-\frac{7t}{d2^i}} \\
&\leq 4 \cdot 2^{-\frac{7t}{d2^k}} \\
&= 4 \cdot 2^{-\frac{7\gamma^7 t}{d}},
\end{aligned}$$

where the first equation follows from Equation (37), the second equation follows from the fact that d is a large enough constant, the third equation follows from basic arithmetics, the forth equation follows from the fact that the series is dominant by a geometrical series, and the latter equation follows from the definition of k.

In this case, the number of chunks with hash collisions in $S_0, S_1, \ldots, S_k$, is at most

$$\sum_{i=0}^{k}\left(2\gamma|S_i| + \frac{\gamma}{2^i}|S|\right) \leq 2\gamma|S| + 2\gamma|S| = 4\gamma|S| \ .$$

Therefore, after receiving $\frac{t}{2^k}$ bits from Bob, at most 4γ fraction of the messages with hash sent by Alice have a collision. This implies that the number of messages with hash collisions sent by Alice, after receiving $\frac{t}{2^k}$ bits from Bob, is at most $4\gamma\Delta a$.

To finish the proof it is suffice to show that the number of messages with hash collisions sent by Alice in this regime, before receiving $\frac{t}{2^k}$ bits from Bob, is at most $\gamma\Delta a$.

Consider the first $\gamma\Delta a$ rounds where Alice sends a message with hash in the regime , and let r be any round proceeding these rounds where Alice sends a message with a hash. Let t_r denote the number of bits received by Alice throughout the protocol until round r. It suffices to show that $t_r \geq \frac{t}{2^k}$.

To this end, let t' denote the number of bits received by Alice before this regime, and let a' denote the number of rounds in which Alice sent a message with hash before this regime. One the one hand, By Equation (35),

$$U' \geq \gamma^6 U = \frac{\gamma^6(t-t')}{\Delta a} \ .$$

One the other hand, by the definition of U',

$$U' \leq \frac{t_r - t'}{a_r - a'} \leq \frac{t_r - t'}{\gamma\Delta a} \ .$$

Putting those together we have that $t_r - t' \geq \gamma^7(t-t')$, and hence

$$t_r \geq \gamma^7 t - \gamma^7 t' + t' \geq \gamma^7 t = \frac{t}{2^k} \ ,$$

as desired. □

□

□

E Proofs from Section 7

E.1 Proof of Claim 20

Proof. We define Π' to simulate Π with the following changes:

1. If a party receives a message m of length $< \alpha^{-1}$ then it interprets this message as a message from Π.

2. If a party receives a message of the form $(0, m)$ of length $\geq \alpha^{-1}$, then the party interprets this message as receiving m according to the protocol Π.

3. If a party receives a message $(1, m)$ of length $\ell' \geq \alpha^{-1}$ then it does the following: Let t denote the total length of all the messages that were sent so far corresponding to Π. If the message is $1^{4\alpha t + L_0}$ then it terminates the protocol. Otherwise it sends 1^ℓ, where $\ell = \min\{\beta^{-1}\ell', 4\alpha t + L_0\}$.

   Loosely speaking, these messages are "padding" messages, and are appended to the transcript of Π' to ensure that the transcript of Π can be recovered from the first $(1-2\alpha)$-fraction of bits in the transcript of Π'.

4. If the protocol Π instructs the party to send a message m of length $< \alpha^{-1}$ then it simply sends m.

5. If the protocol Π instructs a party to send a message m of length $\geq \alpha^{-1}$, then the party sends $(0, m)$.

6. If the protocol Π instructs the party to terminate then the party sends the message 1^ℓ, where $\ell = \min\{\beta^{-1}\ell', 4\alpha t + L_0\}$, where ℓ' is the length of the received message and t is the total communication of the messages from the original protocol.

We first note that Π' is (α, β)-smooth, since adding a single bit to a message can cause that after a message of length ℓ there will be a message of length at most $\frac{\ell}{2\beta} + 1 \leq \frac{\ell}{\beta}$.

We next bound the communication complexity of Π'. We start with a lower bound. To this end, note that long messages that start with 1, are sent only after the simulation of Π ends. Thus at least $\mathrm{CC}(\Pi)$ bits are communicated prior to that. Moreover, since the protocol ends with a message of the form 1^ℓ for $\ell \geq L_0$, we have that $\mathrm{CC}(\Pi') \geq \mathrm{CC}(\Pi) + L_0$.

We next upper bound the communication complexity of Π'. Since the messages corresponding to Π are padded with an extra bit only if they are longer than α^{-1}, the total length of all the messages corresponding to Π is at most $(1+\alpha)\mathrm{CC}(\Pi)$. Since the length of all the extra messages, except the last message, creates a geometric series, their total length is bounded by $\frac{4\alpha\mathrm{CC}(\Pi)+L_0}{1-\beta}$. Thus the total communication complexity of Π' is

$$\mathrm{CC}(\Pi') \leq (1+\alpha)\mathrm{CC}(\Pi) + (4\alpha\mathrm{CC}(\Pi) + L_0)(1 + \frac{1}{1-\beta}) \leq 1 + 13\alpha\mathrm{CC}(\Pi) + 3L_0 \ .$$

We next bound the round complexity. Since all the extra messages, except one, create a geometric series, there total number is bounded by

$$\log_{\frac{1}{\beta}}(4\alpha\mathrm{CC}(\Pi) + L_0) \leq \log_{\frac{1}{\beta}}(\mathrm{CC}(\Pi)) + \log_{\frac{1}{\beta}} L_0.$$

Thus,

$$R(\Pi') \leq R(\Pi) + \log_{\frac{1}{\beta}}(\mathrm{CC}(\Pi)) + \log_{\frac{1}{\beta}} L_0 + 1 \ .$$

Finally, since the total length corresponding to messages of Π is at most $(1+\alpha)\mathrm{CC}(\Pi)$, and the extra messages are of total length at least $4\alpha\mathrm{CC}(\Pi)$, we have that the total length of messages from the original protocol are at most $\frac{1+\alpha}{1+5\alpha} < 1 - 2\alpha$ fraction of the communication complexity of Π'. Thus, if only the $1-2\alpha$ prefix of Π' was encoded correctly, we will be able to decode the messages of Π correctly.

□

E.2 Proof of Claim 21

Proof. We first verify that ϵ, ϵ' and δ satisfy the conditions of Theorem 4. To prove the asymptotic bound on ϵ we will show that $\frac{\alpha^3}{bd\log d} = \tilde{\Omega}(\alpha^{3+\frac{1}{\alpha'}})$ and $\alpha'\alpha^3\beta = \tilde{\Omega}(\alpha^{3+\frac{1}{\alpha'}})$ as follows

$$\frac{\alpha^3}{bd\log d} = \Omega\left(\frac{\alpha'\alpha^3}{d\log d}\right) = \Omega\left(\frac{\alpha'\alpha^4}{\log\frac{1}{\gamma}\log\left(\frac{\log\frac{1}{\gamma}}{\alpha}\right)}\right) = \Omega\left(\frac{\alpha'\alpha^4}{\log^2\frac{1}{\gamma}\log\frac{1}{\alpha}}\right) = \Omega\left(\frac{\alpha'^3\alpha^4}{\log\frac{1}{\alpha}}\right) = \tilde{\Omega}\left(\alpha^{3+\frac{1}{\alpha'}}\right) \ ,$$

$$\alpha'\alpha^3\beta = \alpha'\alpha^{3+\frac{1}{\alpha'}} = \tilde{\Omega}\left(\alpha^{3+\frac{1}{\alpha'}}\right) \ .$$

We next verify the asymptotic bound on ϵ',

$$\epsilon' = \Omega\left(\frac{\alpha'}{d\log\frac{1}{\beta}}\right) = \Omega\left(\frac{\alpha\alpha'}{\log\frac{1}{\gamma}\log\left(\frac{\log^2\frac{1}{\alpha}}{\alpha^{\frac{1}{\alpha'}}}\right)}\right) = \Omega\left(\frac{\alpha\alpha'}{\log\left(\alpha^{-\frac{4}{\alpha'}}\right)\log\left(\frac{1}{\alpha^{\frac{1}{2\alpha'}}}\right)}\right) = \Omega\left(\frac{\alpha\alpha'^3}{\log^3\frac{1}{\alpha}}\right) = \tilde{\Omega}\left(\alpha\alpha'^3\right) \ .$$

We conclude this item by bounding δ as follows,

$$\delta = \gamma^9 = \frac{\alpha^{\frac{36}{\alpha'}}}{2^{360}} \leq \alpha^{\frac{36}{\alpha'}+360} = \alpha^{O(\frac{1}{\alpha'})} .$$

Items 2, 3 and 6 follow trivially. To prove Item 4 it suffices to show that $\beta \leq \frac{1}{5\alpha d^2}$, indeed,

$$\frac{1}{5\alpha d^2} = \frac{\alpha}{5\log^2\frac{1}{\gamma}} = \frac{\alpha}{5\left(\frac{4}{\alpha'}\log\frac{1}{\alpha}+40\right)^2} \geq \frac{\alpha}{5\left(\frac{8}{\alpha'}\log\frac{1}{\alpha}\right)^2} = \frac{\alpha\alpha'^2}{320\log^2\frac{1}{\alpha}} \geq \frac{\alpha^{\frac{1}{\alpha'}}}{320\log^2\frac{1}{\alpha}} = \beta .$$

To prove Item 5 it is suffice to show that $\gamma \leq \frac{1}{d}$. Note that $\gamma < \frac{\alpha}{2\log\frac{1}{\alpha}}$, and thus $\gamma\log\frac{1}{\gamma} \leq \alpha$, which in turn imply $\gamma \leq \frac{\alpha}{\log\frac{1}{\gamma}} = \frac{1}{d}$.

To prove Item 7 we will show that $630\gamma, \frac{72\epsilon}{\alpha^2}, 700d\gamma, 80d\epsilon \leq \frac{1}{4}\alpha\beta$. Indeed,

$$630\gamma \leq \frac{\alpha^{1+\frac{1}{\alpha'}}}{720\log^2\frac{1}{\alpha}} = \frac{1}{4}\alpha\beta ,$$

$$\frac{72\epsilon}{\alpha^2} \leq \frac{1}{4}\alpha\beta ,$$

$$700d\gamma = \frac{700\gamma\log\frac{1}{\gamma}}{\alpha} \leq \frac{700\gamma^{\frac{3}{4}}}{\alpha} = \frac{700\alpha^{\frac{3}{\alpha'}-1}}{2^{30}} \leq \frac{\alpha^{1+\frac{1}{\alpha'}}}{720\log^2\frac{1}{\alpha}} = \frac{1}{4}\alpha\beta ,$$

$$80d\epsilon = \frac{\log\frac{1}{\gamma}\alpha'\alpha^2\beta}{4} \leq \alpha^2\log\frac{1}{\alpha}\beta \leq \frac{\alpha\beta}{4} .$$

To prove Item8 first note that $1812d\log\frac{1}{\beta}\epsilon' = \alpha' \leq \frac{1}{\log\frac{2}{\alpha'}}$. Moreover, since $906d\log\frac{1}{\beta}\epsilon' = \frac{1}{2}\alpha'$ it remain to show the following,

$$90600d\log\frac{1}{\beta}\gamma = \frac{90600\gamma\log\frac{1}{\gamma}\log\left(\frac{245\log^2\frac{1}{\alpha}}{\alpha^{\frac{1}{\alpha'}}}\right)}{\alpha} \leq \frac{90600\gamma^{\frac{3}{4}}\log\left(\frac{1}{\alpha^{\frac{3}{\alpha'}}}\right)}{\alpha} = \frac{3\cdot 90600\log\frac{1}{\alpha}}{2^{30}\alpha\alpha'}\cdot\alpha^{\frac{3}{\alpha'}} \leq \frac{\alpha^{\frac{1}{\alpha'}}}{\alpha'} \leq \alpha' .$$

Finally, to prove Item 9 we first note that

$$\frac{1}{(10\alpha^{-1}+10)L_0} \geq \frac{\alpha}{20L_0} = \frac{\alpha^3}{10000C\log d} \geq \frac{\alpha^5}{d} \geq \frac{\alpha^6}{\log\frac{1}{\gamma}} \geq \gamma^7 \geq \delta .$$

For the second part we first need to verify the following,

$$\frac{\gamma}{6d\log\left(\frac{120d}{\gamma}\right)} \geq \frac{\gamma^2}{720d^2} = \frac{\gamma^2\alpha^2}{720\log^2\frac{1}{\gamma}} \geq \frac{\gamma^2\cdot\gamma^2}{\gamma^{-1}\cdot(\frac{1}{\gamma})^2} = \gamma^6 \geq \delta . \tag{38}$$

To prove the second part of Item 9 we consider $x \geq \frac{1}{\delta}$ and show that $2\cdot 2^{-\delta x} \geq \frac{60d}{\gamma}\cdot 2^{-\frac{\gamma}{3d}x} + \gamma^{-17}\cdot 2^{-\frac{3}{2}\gamma^8 x}$ by proving the following two Equations (39) and (40), as follows,

$$\frac{120d}{\gamma}\cdot 2^{-\frac{\gamma}{3d}x} = \frac{120d}{\gamma}\cdot 2^{-\frac{\gamma}{6d}x-\frac{\gamma}{6d}x} \leq \frac{120d}{\gamma}\cdot 2^{-\frac{\gamma}{6d\delta}}\cdot 2^{-\frac{\gamma}{6d}x} \leq \frac{120d}{\gamma}\cdot 2^{-\log\left(\frac{120d}{\gamma}\right)}2^{-\delta\log\left(\frac{120d}{\gamma}\right)x} \leq 2^{-\delta x} , \tag{39}$$

where the third equation follows from Equation (38).

$$\gamma^{-17}\cdot 2^{-\frac{3}{2}\gamma^8 x} \leq \gamma^{-17}\cdot 2^{-\frac{3}{4}\gamma^8 x-\frac{3}{4}\gamma^8 x} \leq \gamma^{-17}\cdot 2^{-\frac{3}{4}\frac{\gamma^8}{\delta}}\cdot 2^{-\frac{3}{4}\gamma^8 x} \leq 2^{-\gamma^9 x} \leq 2^{-\delta x} . \tag{40}$$

□

E.3 Proof of Lemma 22

Proof. Fix any adversary $\mathcal{A}$ for Π' that corrupts at most ϵ-fraction of the bits of Π' and such that at most ϵ'-fraction of the messages are α^2-corrupted.[23].

The fact that $\mathrm{CC}(\Pi'_{\mathcal{A}}) \geq t_{\min}$, and the fact that S is a probabilistically polynomial time oracle machine (where $\Pi' = (S^A, S^B)$), follows from the fact that each reduction in the definition of Π' is both efficient and only increases the communication complexity, as shown by Lemma 6, Claim 20, Theorem 9, Theorem 17 and Theorem 19. It remains to bound the communication complexity, round complexity and to prove the correctness guarantee.

By Theorem 19, in order to prevent the decoder from decoding a message correctly, the adversary needs to corrupt at least α^2-fraction of the message. Recall that the volume of corrupted messages is defined to be the sum of the lengths of corrupted messages, where the length of each corrupted message is defined be to the maximum between the length of the original message and the length of its corrupted version (see Definition 8). Thus, we can convert $\mathcal{A}'$ to an adversary $\mathcal{A}_{\mathsf{rand}}$, corresponding to protocol Π_{rand}, where the volume of messages corrupted by $\mathcal{A}_{\mathsf{rand}}$ is at most

$$\frac{\epsilon}{\alpha^2} \cdot \mathrm{CC}(\Pi'_{\mathcal{A}}) \leq \frac{2\epsilon}{\alpha^2} \cdot \mathrm{CC}(\Pi_{\mathsf{rand},\mathcal{A}_{\mathsf{rand}}}),$$

where the latter inequality follows from our assumption that for every message m, $|\mathrm{Enc}(m)| \leq 2|m|$.

Moreover, the total number of corrupted messages is bounded by

$$\min\left\{\epsilon \mathrm{CC}(\Pi'_{\mathcal{A}}), \epsilon' R(\Pi'_{\mathcal{A}})\right\} \leq \min\left\{2\epsilon \mathrm{CC}(\Pi_{\mathsf{rand},\mathcal{A}_{\mathsf{rand}}}), \epsilon' R(\Pi_{\mathsf{rand},\mathcal{A}_{\mathsf{rand}}})\right\}.$$

In what follows, we denote

$$\epsilon_{\mathsf{rand}} = \frac{2\epsilon}{\alpha^2} \quad \text{and} \quad \epsilon'_{\mathsf{rand}} = \epsilon'.$$

This, together with Item 6 of Claim 21, implies that $\epsilon_{\mathsf{rand}} \leq \frac{\alpha}{bd \log d}$. Thus by Theorem 17, there exists an adversary $\mathcal{A}_{\mathsf{ideal}}$ for Π_{ideal}, that corrupts at most e'_{rand} messages of total volume at most $e_{\mathsf{rand}} = 2\epsilon_{\mathsf{rand}} \mathrm{CC}(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}})$, where

$$e'_{\mathsf{rand}} = \min\left\{\epsilon \mathrm{CC}(\Pi'_{\mathcal{A}}), \epsilon'_{\mathsf{rand}} R(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) + 2\epsilon'_{\mathsf{rand}} \log_b \mathrm{CC}(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}})\right\} ,$$

and where

$$\epsilon \mathrm{CC}(\Pi'_{\mathcal{A}}) = \frac{\alpha^2}{2} \epsilon_{\mathsf{rand}} \mathrm{CC}(\Pi'_{\mathcal{A}}) \leq \alpha^2 \epsilon_{\mathsf{rand}} \mathrm{CC}(\Pi_{\mathsf{rand},\mathcal{A}_{\mathsf{rand}}}) \leq 2\alpha^2 \epsilon_{\mathsf{rand}} \mathrm{CC}(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) ,$$

where the last inequality follows from Theorem 9 together with the fact that $2600C\alpha \leq 1$. Moreover, by Theorem 17, the adversary $\mathcal{A}_{\mathsf{ideal}}$ chooses the hash collisions in a probabilistic manner, and for every t and every r, with probability $\geq 1 - 20 \cdot 2^{-\frac{\gamma}{3d}t}$, the volume of hash collisions, in the first t bits of $\Pi_{\mathsf{rand},\mathcal{A}_{\mathsf{rand}}}$, is at most $35\gamma t$, and with probability $\geq 1 - 80r \cdot 2^{-7\gamma^8 r}$, the number of rounds with hash collisions, in the first r rounds of $\Pi_{\mathsf{rand},\mathcal{A}_{\mathsf{rand}}}$, is at most $100\gamma r$.

For every t denote by G_t the event that for every $t' \geq t$, the total volume of messages with hash collisions in the first t' bits of $\Pi_{\mathsf{rand},\mathcal{A}_{\mathsf{rand}}}$ is at most $35\gamma t'$. By the union bound, for every t, the probability of event G_t is at least

$$1 - \sum_{t'=t}^{\infty} 20 \cdot 2^{-\frac{\gamma}{3d}t'} = 1 - \frac{20 \cdot 2^{-\frac{\gamma}{3d}t}}{1 - 2^{-\frac{\gamma}{3d}}} \geq 1 - \frac{20 \cdot 2^{-\frac{\gamma}{3d}t}}{\frac{\gamma}{6d}} = 1 - \frac{120d}{\gamma} \cdot 2^{-\frac{\gamma}{3d}t} . \tag{41}$$

[23]Recall that a message is α^2-corrupted if the adversary corrupts at least α^2-fraction of its bits.

where the second equation follows from the fact that $1 - 2^{-x} \geq x/2$ for every $x \leq 1$.

Moreover, for every r, the probability that for every $r' \geq r$ the number of rounds with hash collisions is at most $100\gamma r'$, is at least

$$\begin{aligned}
1 - \sum_{r'=r}^{\infty} 80r' \cdot 2^{-7\gamma^8 r'} &\geq 1 - \sum_{r'=r}^{\infty} 80 \cdot \frac{\log \frac{1}{4\gamma^8}}{2\gamma^8} \cdot 2^{4\gamma^8 r'} \cdot 2^{-7\gamma^8 r'} \\
&\geq 1 - \sum_{r'=r}^{\infty} 80 \cdot \frac{\log \frac{1}{\gamma^8}}{2\gamma^8} \cdot 2^{4\gamma^8 r'} \cdot 2^{-7\gamma^8 r'} \\
&= 1 - \frac{320 \log \frac{1}{\gamma}}{\gamma^8} \cdot \frac{2^{-3\gamma^8 r}}{1 - 2^{-3\gamma^8}} \\
&\geq 1 - \frac{\gamma^{-1}}{\gamma^8} \cdot \frac{2^{-3\gamma^8 r}}{\gamma^8} \\
&= 1 - \gamma^{-17} \cdot 2^{-3\gamma^8 r} \qquad (42)
\end{aligned}$$

To justify the first equation, note that that for every δ and r' it holds that $2^{\delta r'} \geq r'$ for every r' such that $\delta r' \geq \log r'$. Using basic calculus, one can verify that $\delta r' \geq \log r'$ for every $r' \geq \frac{2 \log \frac{1}{\delta}}{\delta}$. Thus, $2^{\delta r'} \geq r'$ for every $r' \geq \frac{2 \log \frac{1}{\delta}}{\delta}$. The first inequality follows from this fact, by setting $\delta = 4\gamma^8$.

Communication Complexity. Let $t_{\mathsf{smooth}} = \mathrm{CC}(\Pi_{\mathsf{smooth}})$. By Lemma 6,

$$t_{\mathsf{smooth}} = (1 + O(\alpha))\mathrm{CC}(\Pi).$$

Let $t_{\mathsf{pad}} = \mathrm{CC}(\Pi_{\mathsf{pad}})$. By Claim 20,

$$t_{\mathsf{pad}} \leq (1 + 13\alpha)t_{\mathsf{smooth}} + 3L_0 = (1 + O(\alpha))\mathrm{CC}(\Pi) + 3L_0 \,.$$

Let $c_\alpha = (1 + \tilde{O}(\alpha)) \leq 3$ to be determined later. Define

$$t'_0 \triangleq \frac{c_\alpha t_{\mathsf{pad}}}{1 - \alpha} = (1 + \tilde{O}(\alpha))\mathrm{CC}(\Pi) + 10L_0$$

and

$$t_0 \triangleq (1 + \alpha)(t'_0 - 10L_0) = (1 + \tilde{O}(\alpha))\mathrm{CC}(\Pi) \,.$$

Fix any $t \geq t_0$. First consider the case where $t < t'_0$. In this case,

$$t'_0 > t > t_0 = (1 + \alpha)(t'_0 - 10L_0)$$

which implies that

$$t < t'_0 < (10\alpha^{-1} + 10)L_0,$$

and in turn implies that

$$2 \cdot 2^{-\frac{t}{(10\alpha^{-1}+10)L_0}} \geq 1.$$

Hence, it trivially hold that

$$\Pr[\mathrm{CC}(\Pi'_{\mathcal{A}}) \geq t] < 2 \cdot 2^{-\frac{t}{(10\alpha^{-1}+10)L_0}}.$$

By (the first part of) Item 9 of Claim 21, we get the desired communication complexity bound.

We next consider the case where $t \geq t'_0$. Let $t_{\mathsf{ideal}} = \frac{t}{c_\alpha}$. By Equation (41), Item 9 of Claim 21, and the fact that $t_{\mathsf{ideal}} \geq \frac{t}{3}$, it suffices to show that if for every $t' \geq t_{\mathsf{ideal}}$ the total volume of hash collisions in the first t' bits of Π_{ideal} is at most $35\gamma t'$, then $\mathrm{CC}(\Pi'_{\mathcal{A}}) \leq t$.

To this end, we first show that $\mathrm{CC}(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) \leq t_{\mathsf{ideal}}$. By our assumption, the fraction of volume of messages with hash collisions or with adversarial corruption in $\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}$ is at most $35\gamma + 2\epsilon_{\mathsf{rand}}$. Thus, By Theorem 9

$$\mathrm{CC}(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) \leq t_{\mathsf{pad}} + 18\beta^{-1}(35\gamma + 2\epsilon_{\mathsf{rand}})\mathrm{CC}(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) + 20d\beta^{-1}(35\gamma + 2\alpha^2\epsilon_{\mathsf{rand}})\mathrm{CC}(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) \ .$$

Hence, by Item 7 of Claim 21,

$$\mathrm{CC}(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) \leq \frac{t_{\mathsf{pad}}}{1 - 18\beta^{-1}(35\gamma + 2\epsilon_{\mathsf{rand}}) - 20d\beta^{-1}(35\gamma + 2\alpha^2\epsilon_{\mathsf{rand}})} \leq \frac{t_{\mathsf{pad}}}{1-\alpha} = \frac{t'_0}{c_\alpha} \leq t_{\mathsf{ideal}} \ ,$$

as desired.

By Theorem 17, $\mathrm{CC}(\Pi_{\mathsf{rand},\mathcal{A}_{\mathsf{rand}}}) \leq (1 + O(\alpha))t_{\mathsf{ideal}}$. Thus, by Theorem 19, we have that

$$\mathrm{CC}(\Pi'_{\mathcal{A}'}) \leq (1 + \tilde{O}(\alpha))\mathrm{CC}(\Pi_{\mathsf{rand},\mathcal{A}_{\mathsf{rand}}}) = (1 + \tilde{O}(\alpha))t_{\mathsf{ideal}} \triangleq c_\alpha t_{\mathsf{ideal}} \ .$$

The result follows from the fact that $t = c_\alpha t_{\mathsf{ideal}}$.

Correctness We assume that for every $t' \geq t_{\mathsf{ideal}}$ the total volume of hash collisions in the first t' bits of Π_{ideal} is at most $35\gamma t'$, where $t_{\mathsf{pad}}, t_{\mathsf{ideal}}$ and t_{rand}, are defined as above. It suffices to assume that $\mathrm{CC}(\Pi'_{\mathcal{A}}) \geq t$ and show that $\Pi_{\mathcal{A}}$ decodes Π correctly.

Since $\mathrm{CC}(\Pi'_{\mathcal{A}}) \geq t$, by the same argument as above, we have that $\mathrm{CC}(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) \geq t_{\mathsf{ideal}}$ and thus the total volume of messages with hash collisions is at most $35\gamma\mathrm{CC}(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}})$.

By Theorem 9, the parties output transcripts of size $\leq \mathrm{CC}(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}})$, and the first N bits of the transcript are consistent with Π_{pad}, for

$$\begin{aligned} N &= \mathrm{CC}(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) - 18\beta^{-1}(35\gamma + 2\epsilon_{\mathsf{rand}})\mathrm{CC}(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) - 20d\beta^{-1}(35\gamma + 2\alpha^2\epsilon_{\mathsf{rand}})\mathrm{CC}(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) \\ &\geq (1-\alpha)\mathrm{CC}(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) \ , \end{aligned}$$

where the inequality follows from Item 7 of Claim 21.

Thus, by Claim 20,

$$\mathsf{Output}(\Pi'_{\mathcal{A}}) = \mathsf{Output}(\Pi_{\mathsf{pad}}) = \mathsf{Trans}(\Pi) \ ,$$

as required.

Round Complexity Let $r_{\mathsf{smooth}} = \mathrm{CC}(\Pi_{\mathsf{smooth}})$. By Lemma 6

$$r_{\mathsf{smooth}} \leq R(\Pi) \cdot (1 + 8\log_{4\beta}\alpha) + 4\log_{\frac{1}{4\beta}} \cdot \mathrm{CC}(\Pi) + 4 = (1 + O(\alpha'))R(\Pi) + O\left(\alpha' \log \mathrm{CC}(\Pi) + 1\right) \ .$$

Let $r_{\mathsf{pad}} = \mathrm{R}(\Pi_{\mathsf{pad}})$, thus by Claim 20,

$$r_{\mathsf{pad}} \leq r_{\mathsf{smooth}} + \log_{\frac{1}{\beta}} \mathrm{CC}(\Pi) + \log_{\frac{1}{\beta}} L_0 + 1 = (1 + O(\alpha'))R(\Pi) + O\left(\alpha' \log \mathrm{CC}(\Pi) + 1\right) \ .$$

We define

$$r_0 = \frac{r_{\mathsf{pad}} + 1812d\log\frac{1}{\beta}\epsilon'_{\mathsf{rand}} \cdot \log_b t_0}{1 - (100\gamma + \epsilon'_{\mathsf{rand}}) \cdot 906d\log\frac{1}{\beta}} + 4\log_b t_0 \ ,$$

and consider $r \geq r_0$. By Item 8 of Claim 21 we have that

$$r_0 = (1 + O(\alpha'))R(\Pi) + O\left(\frac{1}{\log \frac{2}{\alpha'}} \log \mathrm{CC}(\Pi) + 1\right) ,$$

as required.

First, we consider the case where $r \geq t_0$. In this case, by our bound on the communication complexity we have that

$$\Pr\left[R(\Pi'_{\mathcal{A}}) \geq r\right] \leq \Pr\left[\mathrm{CC}(\Pi'_{\mathcal{A}}) \geq r\right] \leq 2 \cdot 2^{-\delta r} .$$

From now on we will consider the case where that $r < t_0$ and let $r_{\mathsf{ideal}} = r - 2\log_b t_0$. Since $r \geq r_0$ we have that $r_{\mathsf{ideal}} \geq \frac{r}{2}$.

We assume that for every $r' \geq r_{\mathsf{ideal}}$ the number of rounds with hash collisions in the first r' bits of Π_{ideal} is at most $100\gamma r'$ and that $\mathrm{CC}(\Pi'_{\mathcal{A}}) \leq t_0$. This suffices since, by the communication bound and Equation 42, the probability that the above does not hold is bounded by

$$\min\left\{1, \frac{60d}{\gamma} \cdot 2^{-\frac{\gamma}{3d} t_0} + \gamma^{-17} \cdot 2^{-3\gamma^8 r_{\mathsf{ideal}}}\right\} \leq \min\left\{1, \frac{60d}{\gamma} \cdot 2^{-\frac{\gamma}{3d} r} + \gamma^{-17} \cdot 2^{-\frac{3}{2}\gamma^8 r}\right\} \leq 2 \cdot 2^{\delta t} ,$$

where the first inequality follows from the fact that $t_0 \geq r$ and $r_{\mathsf{ideal}} \geq \frac{r}{2}$, and the second inequality follows from Item 9 of Claim 21.

We next show that $R(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) \leq r_{\mathsf{ideal}}$. By our assumption, the number of rounds with hash collisions in $\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}$ is at most $100\gamma R(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}})$. Moreover, by Theorem 17, the number of rounds with channel corruptions is at most $\epsilon'_{\mathsf{rand}}(R(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) + 2\log_b t)$. Thus, By Theorem 9

$$R(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) \leq r_{\mathsf{pad}} + 906d \log \frac{1}{\beta} \left(100\gamma R(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) + \epsilon'_{\mathsf{rand}}(R(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) + 2\log_b t_0)\right) .$$

Hence,

$$R(\Pi_{\mathsf{ideal},\mathcal{A}_{\mathsf{ideal}}}) \leq \frac{r_{\mathsf{pad}} + 1812d \log \frac{1}{\beta} \epsilon'_{\mathsf{rand}} \cdot \log_b t}{1 - (100\gamma + \epsilon'_{\mathsf{rand}}) \cdot 906d \log \frac{1}{\beta}} = r_0 - 4\log_b t \leq r - 2\log_b t_0 = r_{\mathsf{ideal}} ,$$

as desired.

Thus, by Theorem 17,

$$R(\Pi'_{\mathcal{A}}) = R(\Pi'_{\mathsf{rand},\mathcal{A}_{\mathsf{rand}}}) \leq r_{\mathsf{ideal}} + 2\log_b t = r .$$

□

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il