Interactive Coding with Small Memory

Klim Efremenko* Bernhard Haeupler† Yael Tauman Kalai‡ Gillat Kol§
Nicolas Resch¶ Raghuvansh R. Saxena‖

Abstract

In this work, we design an *interactive coding scheme* that converts any two party interactive protocol Π into another interactive protocol Π', such that even if errors are introduced during the execution of Π', the parties are able to determine what the outcome of running Π would be in an error-free setting.

Importantly, our scheme *preserves the space complexity* of the protocol, in addition to the communication and computational complexities. Specifically, if the protocol Π has communication complexity T, computational complexity t, and space complexity s, the resulting protocol Π' is resilient to a constant $\epsilon > 0$ fraction of adversarial errors, and has communication complexity approaching T as ϵ approaches 0, computational complexity $\text{poly}(t)$, and space complexity $\mathcal{O}(s \log T)$.

Prior to this work, all known interactive coding schemes required the parties to use at least $\Omega(T)$ space, as the parties were required to remember the transcript of the conversation thus far, or considered weaker error models.

1 Introduction

1.1 Interactive Coding Theory

Background. In the well-studied field of coding theory, which dates back to the seminal work of Shannon [24], researchers attempt to understand the fundamental limits on the transfer of information imposed by unreliable communication channels. Most work in this regime focused on the one-way model of communication, where one party (henceforth referred to as Alice) wishes to send a single message to a second party (Bob). While classical coding schemes have found innumerable applications both in theory and in practice, modern systems, which are often very interactive, have motivated the development of radically new coding schemes.

Motivated by this scenario, Schulman [23] initiated the study of the following model of interactive communication over a noisy channel. There are two parties who wish to carry out a conversation. The additional wrinkle: the channel through which the parties communicate is now unreliable, and may change some of the sent symbols. Therefore, the goal is to transform the original protocol into a new protocol which is still guaranteed to correctly determine the outcome of the original protocol (or, say, succeed with high probability), even if some errors are introduced into the conversation. The new protocol is referred to as a *robust simulation* of the original protocol. In the literature, errors may be random or adversarial, and in our work, we consider the most general adversarial error model, so our results can be applied in all other (weaker) error models.

Resource-efficient interactive coding. Schulman's breakthrough works in the 1990's [21, 22, 23] already showed that every protocol can be robustly simulated by a protocol that only incurs a constant multiplicative overhead in the *communication complexity*, even in the case that an adversary is allowed to corrupt a constant fraction of the total communication. It would be another twenty years before Brakerski and Kalai [1] show that the robust simulation can also be made *computationally efficient*. That is, the running time of the two communicating parties in the simulation protocol is polynomial (or even linear [2]) in the running time of the parties in the original protocol.

*Ben Gurion University. Supported by the Israel Science Foundation (ISF) through grant No. 1456/18 and European Research Council Grant number: 949707.

†ETHZ & Carnegie Mellon University. Supported in part by NSF grants CCF-1814603, CCF-1910588, NSF CAREER award CCF-1750808, a Sloan Research Fellowship, and the European Union's Horizon 2020 ERC grant 949272.

‡Microsoft Research & MIT.

§Princeton University. Supported by a National Science Foundation CAREER award CCF-1750443 and by a BSF grant No. 2018325.

¶University of Amsterdam.

‖Microsoft Research.

Constructing interactive coding schemes that preserve the communication and time complexities of the original protocol (at least up to polynomial factors) is an important step towards making interactive coding practical. In fact, as in the case of (one-way) classical codes, we wish for the coding layer to be as "seamless" as possible and for the simulation to preserve as many of the properties of the original protocol as possible.

In this work, we take another step towards this vision and show that the space complexity can also be preserved. Specifically, we show how to convert any two party protocol into one that is resilient to constant fraction of adversarial errors, while preserving the communication complexity (up to constant factors), the time complexity (up to polynomial factors), and preserving the space complexity (up to logarithmic factors).

We note that all prior works on robust simulation blow up the space complexity of the parties, by forcing them to save in memory the entire transcript of the conversation thus far[1]. Storing the entire transcript can be wasteful, or even impractical for very long protocols, and therefore it is beneficial to have noise-resilient communication protocols which are more space-efficient.

1.2 Our Result Our main result is a time-efficient and space-efficient interactive coding scheme with rate approaching 1 that is resilient to a constant fraction of adversarial errors.

THEOREM 1.1. (INFORMAL STATEMENT; CF. THEOREM 3.2) *Let Π be a protocol of length T, computational complexity t, and space s. Let $\epsilon > 0$ be a small enough constant. There exists a protocol Π' that robustly simulates Π with probability $1 - \frac{1}{T}$ in the presence of an ϵ fraction of adversarial errors and has length approaching T as ϵ approaches 0, computational complexity $\mathrm{poly}(t)$, and space complexity $\mathcal{O}(s \log T)$.*

We mention that our work is essentially a compiler, taking an interactive coding scheme $\mathcal{C}$ (that may not have small space overhead), and outputting a new scheme $\mathcal{C}'$ with a small space overhead. The scheme $\mathcal{C}'$ applies the original scheme $\mathcal{C}$ to blocks of consecutive rounds of Π, and combines the transcripts of Π for these blocks in a space-efficient manner. Since our scheme $\mathcal{C}'$ can be viewed as an efficient oracle machine that uses the original scheme $\mathcal{C}$ as an oracle, it preserves the running time and other properties of the original scheme $\mathcal{C}$. For instance, since in Theorem 3.2, we apply our compiler on the coding scheme $\mathcal{C}$ of [8] that has rate approaching 1, we are able to show that our scheme $\mathcal{C}'$ has rate approaching 1. One can also apply our compiler to other interactive coding schemes to get additional results, *e.g.*, applying it to the scheme of [2] would give a scheme $\mathcal{C}'$ that has an almost linear running time (but a rate that is a constant bounded away from 1).

We believe that the multiplicative $\mathcal{O}(\log T)$ blowup in the space complexity in Theorem 1.1 is necessary when converting protocols into noise resilient ones. Proving or disproving this conjecture is left as an open problem. We mention that a similar blowup appeared in [10][2] (see below for further discussion). At least in the case of [10], proving that this blowup is necessary would require proving new circuit lower bounds.

Our model. We finish this section with a discussion of our specific communication model, noting that our ideas would extend to other models as well. For us, a communication protocol starts with the parties having an input and a memory of s bits and proceeds in rounds: In each round, *each* party sends a bit (which can be an arbitrary[3] function of their input and current memory state) to the other party over the channel. Upon receiving a (possibly flipped) bit from the channel, each party updates their memory state to a new memory state (that is an arbitrary function of their received bit, their input, and their current memory state) and continue executing the remaining rounds. After all the rounds have been executed, the parties use their input and memory state to compute an output value. We mention that in the protocol Π' in Theorem 1.1, the functions that are used to compute the sent bits and update the memory are efficient in terms of time and space, given oracle access to the functions of Π.

We also mention that Theorem 1.1 is shown in the most general fully adaptive adversarial error model. Specifically, the adversary is assumed to know all inputs, and, at any point in the execution of the protocol, the adversary sees all the messages that have been communicated by the parties and all the random strings used by them till that point. For a formal definition, see Section 3.2.

[1] We note that one exception is the work of [7], which is a followup to an earlier version of this work [18], and which we elaborate on in Section 1.3.

[2] In [10], this blowup manifests itself as a blowup in the circuit size. As in our case, the blowup occurs as the simulation needs to remember geometrically spaced locations, called "meeting points", in the original protocol (see Section 2).

[3] In particular, computing this function can take more than s bits of memory.

1.3 Prior Work

Interactive coding. The problem of communicating over a noisy channel was originally formulated by Schulman [21, 22, 23], and since then, there have been numerous works improving various aspects of the resulting robust protocol, such as the communication rate, the error rate, the time complexity, *etc.* [14, 6, 2, 3, 19, 17, 4, 15, 12, 13, 9, 5, to cite a few]. See [11] for an excellent survey.

Space bounded coding. The question of interactive coding with bounded space was first considered in the unpublished manuscript [18][4], which we consider to be an earlier version of this work. Building on ideas from [18], the space bounded interactive coding question was studied in weaker error models: The work of [7] gives an interactive coding scheme that is space efficient (*i.e.*, incurs only logarithmic overhead to the space complexity, similarly to our work) and also achieves the (conjectured) optimal rate as ϵ approaches 0. However, unlike our setting, the scheme of [7] assumes an *oblivious* adversary who makes all its corruption decisions in advance. Specifically, the adversary decides what rounds to corrupt and what to corrupt them to, without seeing the randomness of the parties or the communicated transcript.

We note that assuming an oblivious adversary greatly simplifies the question because the parties can quickly detect when the adversary corrupts the communication[5]. Therefore, they will not add wrong symbols to their local view of the transcript, and in turn, will never need to rewind to a previous version of this transcript before the errors occurred. A space bounded interactive coding scheme follows, as the parties only need to remember the current memory state of the original protocol Π, which is guaranteed to be correct.

Resilient circuits. Another work closely related to ours is the recent work on constructing error resilient circuits [10], which translates the problem of constructing error resilient circuits to the problem of constructing space bounded interactive coding schemes for a non-standard communication-like model (specifically, the model of DAG-protocols with rectangular correctness). Their model is incomparable to the model we consider, with several differences. For example, on the one hand, their model can be seen as a version of the *feedback* model, where the parties know what the symbols they sent were corrupted to, and thus have more information, but on the other hand, their work gives the adversary the power to tamper with the memory of the parties (in addition to the communication), and thus makes his task easier.

2 High-Level Intuition

We now give an informal overview of our interactive coding scheme.

2.1 The Rewind-If-Error Framework The starting point of our interactive coding scheme is the rewind-if-error framework of [21]. Let Π be the noiseless protocol being simulated and let T be the number of rounds in Π. In this framework, the protocol Π is divided into constant-sized blocks[6] and simulated block by block. In more detail, the simulation consists of iterations, with each iteration having a simulation phase and a check phase. In the simulation phase, the parties simulate one block of Π to obtain a transcript for this block. This transcript is appended to the transcript of all the previous blocks and then checked for correctness, by exchanging a hash of the transcript (say), in the check phase. If the check passes, *i.e.*, if the transcript is correct, then parties continue to simulate the next block of Π. If not, the parties "rewind" their transcripts in an attempt to remove the erroneous blocks.

The classic rewind-if-error framework described above suffers from a high space complexity. Indeed, as explained above, when parties run the simulation, they record the entire transcript of Π so far (using $\Omega(T)$ space) even if a noiseless execution of Π can be done using a lot less space, say s. The reason the parties record the entire transcript is threefold: (1) To determine which symbols to send in the next block. (2) To rewind to a previous correct prefix in case errors are detected in the check phase. (3) To compute a hash value that allows the parties to check for correctness.

[4] This manuscript, now retracted, is by a subset of the current authors, and can be found in https://arxiv.org/abs/1805.06872v1.

[5] This can be done, for example, by having the parties exchange hashes of the transcript so far and check that the hashes are the same. An oblivious adversary will not be able to cause hash collisions: To make Alice think that Bob has the same hash value when he does not, the adversary needs to know Alice's hash value and change Bob's communicated hash to this value. Note that the fully adaptive adversaries we consider, that have knowledge of the parties' randomness and the communication history, can easily create such collisions.

[6] Our scheme actually uses blocks of size $\mathcal{O}(\log T)$ for reasons to be explained later. See [1] and followup works for other schemes with logarithmically sized blocks.

In order to get a space bounded interactive coding scheme, one needs to perform Items 1 to 3 using space as close to s as possible. Item 1 is the simplest to handle, as instead of storing the entire transcript so far, the parties can simply store the s bit memory state that it leads to (the current memory state). Now, the fact that Π has space complexity s implies that the bits the parties send in the future blocks of Π are a function of this s bit memory state, and we can use this function in our simulation to compute these bits. Handling Items 2 and 3 is much more involved and discussed next.

2.2 Handling Item 2 via Meeting Points Our solution to handle Item 2 is to adapt the "meeting points" approach of [17]. The main idea here is to have the parties store not only the memory state that the current transcript leads to but also the memory state that some (a carefully chosen set) of its prefixes led to. These prefixes, called meeting points, are chosen to be roughly geometrically apart. In other words, if the parties are currently simulating round i of Π, then, for all $z \in [\log T]$, the parties also remember the memory state that a prefix of length (roughly) $i - 2^z$ led to. In fact, remembering these $\log T$ prefixes is the reason why our coding scheme increases the space by a $\log T$ factor.

With this extra memory to store the meeting points, the parties perform the rewinds mentioned in Item 2 by going to the closest meeting point that they have in memory. Then, the next iteration will check if this meeting point is correct. If even this meeting point is found to be incorrect, the parties will rewind to the one before, and so on. Note that as the meeting points are geometrically spaced, the parties never have to rewind more than twice the number of rounds corrupted by the adversary. This is crucial and allows us to handle a constant fraction of adversarial errors, as explained next.

Why remember $\log T$ meeting points? Recall that the reason Theorem 1.1 increases the space by a $\log T$ factor is that the parties remember $\log T$ meeting points. One might wonder if we can get an improvement by simply remembering fewer meeting points. Such an improvement, without any major new ideas, is unlikely, as if the parties remember $o(\log T)$ meeting points, then there exists a length l such that the parties have no meeting point between the rounds $i - 100l$ to $i - l$.

Now, imagine the adversary corrupted all rounds starting from $i - l$ to round i. To fix these corruptions the parties will have to go back to the closest correct meeting point (before round $i - 100l$) and continue the simulation from there. Thus, by inserting l corruptions, the adversary was able to make the parties redo $100l$ rounds, implying that the protocol cannot handle more than a $\frac{1}{100}$ fraction of corruptions. As the constant 100 was arbitrary, it follows that the protocol cannot be resilient to any constant fraction of corruptions.

The variable E. Another way the adversary can cause the parties to waste many rounds with a small number of corruptions is by causing "fake rewinds". Imagine that the parties are currently simulating a round, say i, of Π, and the transcript so far is correct. However, the parties do not remember any meeting points close to i (due to previous rewinds, for example) and the closest meeting point they remember is $i - \Delta$, for some large Δ. Now, if the adversary can insert a small number of corruptions to make the parties believe that their transcript of length i is actually incorrect and send them to point $i - \Delta$, then he has again made them waste many rounds with a small number of corruptions.

To prevent this from happening, we maintain a variable E that helps the parties avoid fake rewinds. At a high level, whenever the parties want to rewind, they will increment E by 1 and only rewind when E reaches Δ, when they also set E back to 0. This means that if the rewinds are actually fake, the adversary needs to insert Δ corruptions, and the previous attack does not work. Finally, we mention that the variable E does slow down the rewind process in case the rewinds are not fake, but if the rewinds are not fake, the adversary already inserted enough corruptions to make the parties' transcripts incorrect, and we can afford the slowdown.

2.2.1 Taxes As is evident from the examples above, the length of the correct transcript remembered by the parties[7] may change by a lot in one iteration of our interactive coding scheme. This complicates our analysis significantly, as traditional approaches of showing that a potential function (which is governed, amongst other things, by the length of the correct transcript) increases in every iteration do not work any more.

As an extreme example, consider the following scenario: The simulation proceeds correctly for the first i iterations and the parties have a correct transcript of length i. At this point, the adversary starts inserting a lot

[7] More formally, the longest prefix of the correct transcript of Π which we currently have stored as a meeting point (*i.e.*, the corresponding memory state is stored).

of corruptions and eventually takes the parties to an iteration $i' > i$ where they are about to forget[8] the meeting point at length i. Assume for simplicity that by the time parties reach round i', they have already forgotten all the meeting points with length smaller than i. Then, before iteration i', the parties have a correct transcript of length i but immediately afterwards, the length decreases to 0, causing a huge drop in the potential in a single iteration.

The way to fix this situation is to note that in order to get this huge drop in potential in a single iteration, the adversary must have inserted a lot of corruptions in many of the previous iterations. Thus, if we can somehow "charge" this decrease in potential to those corruptions, our analysis might still work. We do exactly this, and work with a more involved potential function that has an extra term called Tax. When the adversary tries to insert corruptions hoping to cause a sudden drop in potential after a large number of iterations, we use these corruptions to increase the value of Tax. Eventually, when the drop occurs, we reset Tax to 0 and use this decrease to counter the change in the length of the correct transcript. For more details, see Lemmas 5.0.23 and 5.0.24.

2.3 Handling Item 3 via Chaining Hashes It remains to describe how the parties check if the simulation so far is correct without the knowledge of the transcript. Recall that, if the parties knew the transcript, they could simply exchange a hash of this transcript and if the hashes match, deduce that the transcript is likely to be correct[9]. However, we cannot afford to remember the entire transcript, and in our scheme, the parties only remember the memory state that the transcript leads to.

As a first attempt, one might consider hashing these memory states and checking if the hash values are equal. Unfortunately, this does not quite work as illustrated by the following example (also described in [7]): Let Π be the length $\mathcal{O}(n)$ protocol that computes the Hamming distance of two n-length bit vectors. Such a protocol is possible using $\mathcal{O}(\log n)$ space as the parties can exchange their vectors bit by bit, only counting how many bits so far were different.

This protocol indeed works if there are no errors, but if there are errors, then by corrupting the bits receive by both parties, an adversary can make both the parties wrongly increment the counter[10]. As the counter is essentially all the parties remember, and the fact that both of them wrongly incremented it, means that their values for the counter still agree, the parties will not be able to detect this error by simply exchanging hashes of their memory. More generally, the fact that the two parties have the *same memory state does not imply that this memory state is the correct one.*

Chaining hashes. Our solution is to maintain a "chained" hash of their view of the transcripts *and* the hash seeds, and exchanging this hash to verify correctness. In more details, after the first iteration, the parties only have a small transcript and can remember it in its entirety and compute a hash H_1. Then, after the second iteration, the parties hash the small transcript of this iteration *and* the hash H_1 *and* the hash seed for H_1 to compute a new hash H_2. In the third iteration, the parties would then hash the small transcript of this iteration and H_2 and the hash seed for H_2 to get H_3, and so on.

The reason we hash this way is that it (except with probability polynomially small in T) ensures that, if the hashes used by the parties are $\Omega(\log T)$ in length, then, for all $i \in [T]$, the pair $\mathcal{H}_i = (H_i, \text{seed for } H_i)$ is the same for the parties only if their transcripts so far are the same (our notation for pairs follows the notation in our protocol, see Eq. (4.4)). Put differently, looking at the current hash value and the current hash seed (which can be maintained in $\mathcal{O}(\log T)$ space) allows the parties to check equality of their (much longer) transcripts. To show the statement, we proceed by contradiction. Consider the smallest i such that the $\mathcal{H}_i$ is the same for both the parties but the transcripts at iteration i are not. By our choice of i, we have that either $\mathcal{H}_{i-1}$ is different for the parties, or the transcripts for iteration $i-1$ are the same. As the transcripts at iteration i are not the same, the latter can only happen if the small transcript of iteration i is different.

Now, combine these facts to get that, at iteration i, the values of $\mathcal{H}_i$ are the same for the parties but either

[8] This must happen as the parties do not have enough memory to store all the lengths and they do not know which part of the transcript is uncorrupted.

[9] The reason is that Alice's transcript has the correct symbols communicated in rounds where she transmits, and Bob's transcript has the correct symbols communicated in rounds where he transmits. If the transcripts match, all rounds are correct.

[10] For instance, consider the case where the first bit in the inputs of both parties is 0, but the adversary corrupts the transmission of these first bits to make it sound like they are both 1. In this case, after exchanging the first bits, the counter of both parties will be 1 instead of 0. The reason is that, upon getting the 1 bit from Bob, since her first bit is 0, Alice increments her counter. Bob does the same.

$\mathcal{H}_{i-1}$ or the small transcripts at iteration i are different. This means that the parties hash two different objects with the same seed in iteration i and get the same hash value. As the hashes used by the parties are $\Omega(\log T)$ in length, this can only happen with probability polynomially small in T for any given i. Union bounding over the $\mathcal{O}(T)$ many iterations i finishes the proof.

Blocks of length $\mathcal{O}(\log T)$. The above approach only works if the hashes the parties exchange are of length $\Omega(\log T)$, as otherwise there may be hash collisions and we lose our correctness guarantee. In turn, this means that the length of the check phase in any iteration must be at least $\Omega(\log T)$. With such a long check phase, the only way we can have a constant rate interactive coding scheme that is resilient to a constant fraction of errors is if the simulation phase also simulates a block of length $\Omega(\log T)$ of Π and works even if a constant fraction of its rounds are corrupted.

To get these guarantees, in our simulation phase, we actually simulate a $\Theta(\log T)$-length block of Π using one of the classic (not necessarily space bounded and with only a weak exponential time bound) interactive coding schemes present in the literature. Using a scheme that is not space bounded is okay as the input to the scheme is a protocol of length $\Theta(\log T)$ and thus, even in the worst case, it can only take $\mathcal{O}(\log T)$ space and poly(T) running time, which we can afford.

3 Preliminaries

3.1 Building Blocks

Our protocol can make use of the following objects:

Randomness Efficient Hash Functions. The following lemma is due to [20] (see also [18]):

LEMMA 3.0.1. *There exists a constant K_1 such that the following holds: Let $n, m > 0$ and define $c = K_1 \cdot (m + \log n)$. There exists a family of functions $\{\mathsf{h}_{\mathsf{sd}}\}_{\mathsf{sd}\in\{0,1\}^c}$ mapping $\{0,1\}^n \to \{0,1\}^m$ such that for all $x \neq y \in \{0,1\}^n$, it holds that:*

$$\Pr_{\mathsf{sd}\sim\{0,1\}^c}(\mathsf{h}_{\mathsf{sd}}(x) = \mathsf{h}_{\mathsf{sd}}(y)) \leq 2^{-m/K_1}.$$

Furthermore, the time required to evaluate h, given inputs sd and x is polynomial in n, m.

Error Correcting Codes. We use the following standard result for error-correcting codes (see, *e.g.*, [25, 16]):

LEMMA 3.0.2. *There exists a constant K_2 such that the following holds: For all $n > 0$, there exists a function $\mathsf{ECC}_n : \{0,1\}^n \to \{0,1\}^{K_2 n}$ such that for all $s \neq t \in \{0,1\}^n$, we have*

$$\Delta(\mathsf{ECC}_n(s), \mathsf{ECC}_n(t)) > 0.1 \cdot K_2 n.$$

Here, $\Delta(\cdot,\cdot)$ denotes the Hamming distance. Furthermore, the time required to evaluate ECC_n is polynomial in n.

Note that maximum likelihood decoding for an ECC as above can be performed in time at most exponential in n by brute force.

3.2 Two-Party Communication with Bounded Memory

Recall that our main result is a memory-efficient interactive coding scheme against adversarial noise. Our scheme will be randomized where the parties have access to random coins in each round of the scheme, and the adversary in any given round will know all the random bits sampled by the parties so far, but will not know any random bits they will sample in the future. Specifically, we consider a formalization Alice and Bob are denoted by A and B respectively and where a protocol is defined by a tuple:

$$\Pi = \left(T, s, \{\mathcal{X}^C\}_{C\in\{A,B\}}, \mathcal{Y}, \{\mathsf{msg}_j^C\}_{C\in\{A,B\},j\in[T]}, \{\mathsf{mem}_j^C\}_{C\in\{A,B\},j\in[T]}, \{\mathsf{out}^C\}_{C\in\{A,B\}}\right),$$

where: (1) $T = \|\Pi\|$ is the number of rounds in Π, (2) $s = \mathsf{Sp}(\Pi)$ is the space required by Π, (3) For all $C \in \{A, B\}$, $\mathcal{X}^C$ is the input set of party C, (4) $\mathcal{Y}$ is the output space of the protocol, (5) For all $j \in [T]$ and all $C \in \{A, B\}$, $\mathsf{msg}_j^C : \mathcal{X}^C \times \{0,1\}^s \times \left(\{0,1\}^*\right)^j \to \{0,1\}$, is a function that takes as input the input of party C, the current memory state of party C, and the random bits sampled by party C in the rounds so far, and computes a bit that they will send in this round, (6) For all $j \in [T]$ and all $C \in \{A, B\}$, $\mathsf{mem}_j^C : \mathcal{X}^C \times \{0,1\}^s \times \{0,1\} \times \left(\{0,1\}^*\right)^j \to \{0,1\}^s$, is a function that takes as input the input of party C, the current memory state of party C, the bit received by party

C in round j, and the random bits sampled by party C in the rounds so far, and computes their new memory state. (7) For all $C \in \{A, B\}$, $\mathsf{out}^C : \mathcal{X}^C \times \{0,1\}^s \times \left(\{0,1\}^*\right)^T \to \mathcal{Y}$ is a function that takes as input the input of party C, its final memory state, all their randomness, and computes their output in the protocol.

We say that a protocol Π is deterministic if for all $j \in [T]$ and all $C \in \{A, B\}$, the functions mem_j^C, msg_j^C, and out^C do not depend on their last argument.

Adversaries. Let Π be a protocol as above. An adversary Adv for Π is defined by the tuple $\mathsf{Adv} = \left(\mathsf{Adv}_j^C\right)_{C\in\{A,B\}, j\in[T]}$, where for all $C \in \{A, B\}$ and $j \in [T]$, the function $\mathsf{Adv}_j^C : \mathcal{X}^A \times \mathcal{X}^B \times \left(\{0,1\}^*\right)^j \times \left(\{0,1\}^*\right)^j \to \{0,1\}$ takes as inputs the inputs of the two parties and the randomness sampled by them in the first j rounds and outputs a bit that they will receive in round j. We stress that this function does not depend on the randomness that the parties will sample in the future rounds.

Execution of a protocol. Let Π be a protocol as above and Adv be an adversary for Π. Let $x^A \in \mathcal{X}^A$, $x^B \in \mathcal{X}^B$ be a pair of inputs for Alice and Bob respectively. Similarly, let $\mu_0^A = \mu_0^B = 0^s$ denote the initial memory state of Alice and Bob respectively. For $C \in \{A, B\}$ and $j \in [T]$, let R_j^C denote the random string sampled by party C in round j. We shall use $R_{\le j}^C$ to denote the tuple $\left(R_1^C, \cdots, R_j^C\right)$ and sometimes write $R_{<j}^C$ instead of $R_{\le j-1}^C$. The execution of the protocol Π on inputs x^A, x^B in the presence of Adv proceeds as follows:

At the beginning of the execution, Alice and Bob start with memory states μ_0^A and μ_0^B respectively. In round $j \in [T]$, party $C \in \{A, B\}$ computes the bit $\beta_j^C = \mathsf{msg}_j^C\left(x^C, \mu_{j-1}^C, R_{\le j}^C\right)$ and sends it over the channel. In turn, it receives the $\beta_j'^C = \mathsf{Adv}_j^C\left(x^A, x^B, R_{\le j}^A, R_{\le j}^B\right)$ and then updates its memory state to $\mu_j^C = \mathsf{mem}_j^C\left(x^C, \mu_{j-1}^C, b_j'^C, R_{\le j}^C\right)$ and moves to the next round of the protocol. After T rounds, party C simply outputs $y^C = \mathsf{out}^C\left(x^C, \mu_T^C, R_{\le T}^C\right)$. We define $\Pi_{\mathsf{Adv}}\left(x^A, x^B\right) = \left(y^A, y^B\right)$.

Corruptions. Using the same notation as above, for $C \in \{A, B\}$ and $j \in [T]$, we say that the message to party C is corrupted if the symbol received by party C in round j is different from the symbol sent by the other party in round j. More precisely, we define

$$\mathsf{corr}_j^A(\Pi, \mathsf{Adv}, x^A, x^B) = \mathbb{1}\left(\beta_j^B \neq \beta_j'^A\right) \qquad \text{and} \qquad \mathsf{corr}_j^B(\Pi, \mathsf{Adv}, x^A, x^B) = \mathbb{1}\left(\beta_j^A \neq \beta_j'^B\right).$$

We also define $\mathsf{corr}_j(\cdot) = \mathsf{corr}_j^A(\cdot) + \mathsf{corr}_j^B(\cdot)$ and $\mathsf{corr}(\cdot) = \sum_{j\in[T]} \mathsf{corr}_j(\cdot)$. Observe that all the quantities in the previous two paragraphs are actually random variables that are functions of the randomness sampled by the parties. For $0 \le \epsilon \le 1$, we say that the adversary corrupts at most ϵ fraction of the messages of Π if for all inputs x^A, x^B, it holds that $\mathsf{corr}(\Pi, \mathsf{Adv}, x^A, x^B) \le 2\epsilon T$ almost surely. When we omit writing Adv in our notations above, we mean an adversary that corrupts a 0 fraction of the messages of Π. Observe that in this case, all the quantities mentioned above are determined by Π, x^A, and x^B.

Simulating protocols. Let Π be a deterministic protocol to be simulated[11] and $0 \le \epsilon, p \le 1$ be parameters. Let Π' be a randomized simulation protocol with the same input sets for the parties. We say that the protocol Π' simulates Π with probability p in the presence of an ϵ fraction of errors if for all inputs x^A, x^B and all adversaries Adv for Π' that corrupt at most ϵ fraction of the messages of Π', it holds that:

$$\Pr\left(\Pi'_{\mathsf{Adv}}\left(x^A, x^B\right) = \Pi\left(x^A, x^B\right)\right) \ge p,$$

where the probability is over the randomness sampled by the parties in Π'. We omit writing p when $p = 1$.

REMARK 3.0.1. *Note that for any protocol Π as above, there is an equivalent protocol Π' with the same number of rounds that also satisfies $\mathsf{Sp}(\Pi') \le T + 1$. This is because the parties can simply memorize all the bits they receive and reconstruct the actual memory state "on the fly" using these bits. Thus, we always will assume the bound $\mathsf{Sp}(\Pi') \le T + 1$.*

REMARK 3.0.2. (SPACE COMPLEXITY OF A PROTOCOL) *Note that our definition of $\mathsf{Sp}(\cdot)$ above does not actually take into account the space needed by the parties to compute the functions msg, mem, and out. This is done only to make the definition cleaner and the schemes we describe in this paper do not suffer from a high space complexity.*

[11] We restrict attention to deterministic protocols as a randomized protocol can be simulated by simulating all deterministic protocols in its support.

We finish this section by recalling a well-known interactive coding result, which can be found in [8] (see also [17]).

THEOREM 3.1. *Let* Π *be a deterministic protocol and* $\epsilon > 0$. *There exists a deterministic protocol* Π *that simulates* Π *in the presence of an* ϵ *fraction of errors such that* $\|\Pi'\| \leq \|\Pi\| \cdot \left(1 + \mathcal{O}\left(\sqrt{\epsilon \log(1/\epsilon)}\right)\right)$. *Furthermore, the parties in* Π' *run in time at most exponential in* $\|\Pi\|$ *given oracle access to* Π.

3.3 Formal Statement of Main Result

We are now ready to formally state our main result.

THEOREM 3.2. *Let* Π *be a deterministic protocol with* $T = \|\Pi\|$ *and* $s = \mathsf{Sp}(\Pi)$. *Let* $\epsilon > 0$ *be a small enough constant. There exists a (randomized) protocol* Π' *that simulates* Π *with probability* $1 - \frac{1}{T}$ *in the presence of an* ϵ *fraction of errors such that:*

$$\|\Pi'\| \leq T \cdot \left(1 + \mathcal{O}\left(\sqrt[3]{\epsilon \log(1/\epsilon)}\right)\right) \quad \text{and} \quad \mathsf{Sp}(\Pi') \leq \mathcal{O}(s \log T).$$

Furthermore, the parties in Π' *run in time at most polynomial in* T *given oracle access to* Π.

4 Space Bounded Interactive Coding

The goal of this section is to prove Theorem 3.2. We fix Π and ϵ as in the theorem statement. By increasing $\mathsf{Sp}(\Pi)$ by an additive $\log T$, we can assume without loss of generality that the memory of both the parties in Π contains (at least) the round number that they are executing. Throughout, we use $s = \mathsf{Sp}(\Pi)$ to denote the space required by Π. Let K be a constant much larger than the constants promised by Lemmas 3.0.1 and 3.0.2 and define the parameters:

$$h = 10K \log T \qquad \epsilon' = \frac{\epsilon^{2/3}}{\sqrt[3]{\log(1/\epsilon)}} \qquad B^* = \frac{K^5}{\epsilon'^5} \cdot \log T \tag{4.1}$$

$$M = \frac{T}{B^*} \qquad M' = M \cdot \left(1 + 10^6 \cdot \frac{\epsilon}{\epsilon'}\right) \qquad r = 10Kh \tag{4.2}$$

$$r' = \frac{K^3}{\epsilon'^3} \cdot \log T \gg r \tag{4.3}$$

We also define $\{\mathsf{h}_{\mathsf{sd}}\}$ to be the hash function family promised by Lemma 3.0.1 with $n = 10B^*$ and $m = h$. This family will be used in Line 6 and we will ensure that the input to the hash function is at most $10B^*$ bits in length, and thus can be hashed after being padded appropriately. Observe that $2r$ bits of randomness are sufficient to sample a function from this family and therefore, we will assume $\mathsf{sd} \in \{0,1\}^{2r}$. For all $\mathsf{sd} \in \{0,1\}^{2r}$, we now define the auxiliary function:

$$\mathcal{H}_{\mathsf{sd}}(\cdot) = \mathsf{h}_{\mathsf{sd}}(\cdot) \| \mathsf{sd}. \tag{4.4}$$

Namely, the function $\mathcal{H}$ outputs the output of h concatenated with sd. We shall use this function in our protocol. Finally, we let ECC be as promised by Lemma 3.0.2 with $n = r'$. The error correcting code ECC will be implicitly used in our protocol in Lines 5 and 7, *i.e.*, the messages exchanged by the parties in this line will be encoded using ECC and the receiving party will decode to the closest possible message. We will ensure that message being encoded is at most r' bits and thus ECC can be applied after padding the message appropriately.

4.1 Our Protocol

We now describe the protocol Π' that shows Theorem 3.2. Roughly speaking, in the protocol Π', both Alice and Bob, maintain a pair Z which contains a memory state for the protocol Π being simulated and a hash of the transcript that led to this memory state. We will use $\mathsf{Z.state}$ to denote the memory state and $\mathsf{Z.hash}$ to denote the hash. Recalling our assumption that the memory state of a party Π contains the round number, we get that Z also determines a unique round number $\mathsf{Z.rn}$ in Π where its memory state can happen. As our protocol shall attempt to simulate Π in blocks of length B^*, we define $\mathsf{Z.bn} = \frac{\mathsf{Z.rn}}{B^*}$ to be the block number corresponding to the memory state of Z.

Another important variable in the protocol Π' is the set MP of meeting points. As the protocol Π' may have errors, the parties may sometimes need to rewind to correct those errors, and this is done through the set MP.

More precisely, the set MP is initially empty but as the parties execute Π', they store some values Z in the set MP (see Line 9). If they decide to rewind (Line 17), they always rewind to a pair in MP with the largest bn. Throughout, we adopt the convention that $\max_{\mathsf{M}\in\mathsf{MP}} \mathsf{M.bn} = -1$ if $\mathsf{MP} = \emptyset$. For integers x, y, we also use $\lfloor x \rfloor_y$ to denote the largest multiple of y that is at most x. For example, we have $\lfloor 5 \rfloor_3 = 3$ and $\lfloor 28 \rfloor_7 = 28$.

Finally, recall that our protocol Π' simulates Π in blocks of length B^*. We now describe how this is done in more detail. In Line 3, the parties (implicitly) construct a new protocol, call it Π_{block}, where the parties start from the memory state $\mathsf{Z.state}$ (instead of the default starting state) and execute B^* rounds of Π starting from round $\mathsf{Z.bn}$ with the goal of outputting the length $2B^*$ transcript, *i.e.* the transcript that contains the B^* symbols they sent and the B^* symbols they received. We shall assume that Π is padded with sufficiently many dummy rounds so that Π_{block} is always well defined. The parties then construct a noise resilient version of Π_{block} using Theorem 3.1 with Π_{block} and ϵ' and execute this noise resilient version. By Theorem 3.1, this needs at most $B^* \cdot \left(1 + \mathcal{O}\left(\sqrt{\epsilon' \log(1/\epsilon')}\right)\right) < 2B^*$ rounds of communication. Finally, in Line 10, the parties use the received symbols in the transcript (of length B^*) obtained in Line 3 to update the memory state $\mathsf{Z.state}$ of Π (using the function mem).

We now formally describe our protocol.

Algorithm 1 Alice's side of the space-bounded interactive coding scheme Π'.

Require: Alice starts with an input $x^A \in \mathcal{X}^A$.

1: $\mathsf{Z} \leftarrow \left(\mu_0^A, \perp\right)$, $\mathsf{MP} \leftarrow \emptyset$, $E \leftarrow 0$.
2: **for** $i \in [M']$ **do**
3: Using the input x^A, simulate a block of Π from $\mathsf{Z.state}$ with resilience ε' to get $\sigma \in \{0,1\}^{2B^*}$. As explained above, this is done using the scheme from Theorem 3.1.
4: Sample a uniformly random string $R \in \{0,1\}^r$.
5: Send R and receive R'. Set $\mathsf{sd} = R \| R' \in \{0,1\}^{2r}$.
6: $H \leftarrow \mathcal{H}_{\mathsf{sd}}(\mathsf{Z.hash}, \sigma)$, $b \leftarrow \mathsf{Z.bn}$.
7: Send (H, E, b) and receive (H', E', b').
8: **if** $H = H'$ **and** $E = E' = 0$ **then**
9: $\mathsf{MP} \leftarrow \mathsf{MP} \cup \{\mathsf{Z}\}$.
10: $\mathsf{Z} \leftarrow \left(\text{Use } \sigma \text{ and } x^A \text{ to update } \mathsf{Z.state}, H\right)$.
11: **else if** $H = H'$ **then**
12: $E \leftarrow \max(E - 1, 0)$.
13: **else if** $b \geq b'$ **then**
14: $E \leftarrow E + 1$.
15: **if** $\max_{\mathsf{M}\in\mathsf{MP}} \mathsf{M.bn} \geq b - E$ **then**
16: $E \leftarrow E + \max_{\mathsf{M}\in\mathsf{MP}} \mathsf{M.bn} - b$.
17: If $\mathsf{MP} \neq \emptyset$, set $\mathsf{Z} \leftarrow \arg\max_{\mathsf{M}\in\mathsf{MP}} \mathsf{M.bn}$, breaking ties lexicographically.
18: **end if**
19: **end if**
20: $\mathsf{MP} \leftarrow \{\mathsf{M} \in \mathsf{MP} \mid \exists z \geq 0 : \mathsf{M.bn} = \lfloor \mathsf{Z.bn} \rfloor_{2^z} - 2^z\}$.
21: **end for**
22: Output what Alice would have output in Π if her input was x^A and memory was $\mathsf{Z.state}$.

5 Analysis

5.1 Complexity

We first show that our protocol in Algorithm 1 is not too long and does not require too much memory.

LEMMA 5.0.1. *It holds that:*

$$\|\Pi'\| \leq T \cdot \left(1 + \mathcal{O}\left(\epsilon^{1/3} \cdot \sqrt{\log(1/\epsilon)}\right)\right).$$

Proof. Note that the only communication in Π' is in Lines 3, 5 and 7. These lines are executed in each iteration

and require communication at most $B^* \cdot \left(1 + \mathcal{O}\left(\sqrt{\epsilon' \log(1/\epsilon')}\right)\right)$, Kr', and Kr' respectively. We get:

$$\begin{aligned}
\|\Pi'\| &\leq M' \cdot \left(B^* \cdot \left(1 + \mathcal{O}\left(\sqrt{\epsilon' \log(1/\epsilon')}\right)\right) + 2Kr'\right) \\
\text{(Eq. (4.1))} \quad &\leq \frac{T}{B^*} \cdot \left(1 + 10^6 \cdot \sqrt[3]{\epsilon \log(1/\epsilon)}\right) \cdot \left(B^* \cdot \left(1 + \mathcal{O}\left(\sqrt{\epsilon' \log(1/\epsilon')}\right)\right) + \epsilon' B^*\right) \\
&\leq T \cdot \left(1 + 10^6 \cdot \sqrt[3]{\epsilon \log(1/\epsilon)}\right) \cdot \left(1 + \mathcal{O}\left(\sqrt{\epsilon' \log(1/\epsilon')}\right) + \epsilon'\right) \\
\text{(Eq. (4.1))} \quad &\leq T \cdot \left(1 + 10^6 \cdot \sqrt[3]{\epsilon \log(1/\epsilon)}\right) \cdot \left(1 + \mathcal{O}\left(\sqrt[3]{\epsilon \log(1/\epsilon)}\right) + \epsilon^{2/3}\right) \\
&\leq T \cdot \left(1 + \mathcal{O}\left(\sqrt[3]{\epsilon \log(1/\epsilon)}\right)\right).
\end{aligned}$$

□

LEMMA 5.0.2. *It holds that:*

$$\mathsf{Sp}(\Pi') \leq \mathcal{O}(s \log T).$$

Proof. To bound the space complexity of Π', we simply examine the variables used by Π'. These are the variables Z, MP, E, i, σ, R, R', sd, H, b, H', E', b', and the space needed to run Line 3. Using Remark 3.0.1, the space needed to run Line 3 is at most $2B^*$. Thus, the space need for all variables other than Z and MP is:

$$\mathcal{O}(B^* + \log T + r) = \mathcal{O}(\log T),$$

by Eq. (4.1). To analyze the space needed by MP, we first upper bound $|\mathsf{MP}|$. For this, note that Line 20 ensures that $\mathsf{M.bn} < \mathsf{Z.bn}$ for all $\mathsf{M} \in \mathsf{MP}$. This together with Line 9 ensures (by induction) that all elements $\mathsf{M} \in \mathsf{MP}$ have different values $\mathsf{M.bn}$. Finally, Line 20 also ensures that the number of such values is at most $\log T$ implying that $|\mathsf{MP}| \leq \log T$. Thus, the space needed for the variables Z and MP is

$$\mathcal{O}(\log T) \cdot (s + r) = \mathcal{O}(\log T) \cdot (s + \log T),$$

by Eq. (4.1). Adding, we get:

$$\mathsf{Sp}(\Pi') \leq \mathcal{O}(\log T) \cdot (s + \log T) = \mathcal{O}(s \log T).$$

□

The following observation is due to our choice of parameters in Eq. (4.1).

OBSERVATION 5.0.1. *The runtime of the parties in Algorithm 1 is polynomial in T assuming oracle access to Π.*

5.2 Correctness We now show that the protocol Π' indeed simulates the protocol Π with probability $1 - \frac{1}{T}$ in the presence of an ϵ fraction of errors. For this, fix inputs x^A and x^B for Alice and Bob respectively and also fix an adversary Adv satisfying $\mathsf{corr}(\Pi', \mathsf{Adv}, x^A, x^B) \leq 2\epsilon \cdot \|\Pi'\|$ almost surely. Using Lemma 5.0.1, we have $\mathsf{corr}(\Pi', \mathsf{Adv}, x^A, x^B) \leq 4\epsilon T$. We have to show that:

$$\Pr\left(\Pi'_{\mathsf{Adv}}(x^A, x^B) = \Pi(x^A, x^B)\right) \geq 1 - \frac{1}{T}, \tag{5.5}$$

Next, note that fixing Adv and the inputs implies that all the variables in Algorithm 1 (for both Alice and Bob) are random variables that are functions of the randomness sampled by the parties in Line 5. For $i \in [M']$ and a variable var in Algorithm 1, we will use var_i^A to denote Alice's value of var at the end of iteration i of the loop in Line 2. We will use var_0^A to refer to the value of var at the beginning of the loop. The notations var_i^B and var_0^B are defined analogously. We also define the notation $R_i = \left(R_i^A, R_i^B\right)$ for all $i \in [M']$ and fix R_0 to be a dummy value. We may use the sans-serif letters, *e.g.*, R_i, to emphasize that we are looking at R_i as a random variable instead of a fixed realization.

Finally, observe that for all $i \in [M']$, the values of σ_i^A, σ_i^B are determined by $R_1, \ldots, R_{i-1}$, and the values var_i^A, var_i^B of all other variables are determined by $R_1, \ldots, R_i$.

5.2.1 No Hash Collisions

LEMMA 5.0.3. *Let $i \in [M']$. For all $R_1, \ldots, R_{i-1}$ such that $\left(\mathsf{Z}^A_{i-1}.\mathtt{hash}, \sigma^A_i\right) \neq \left(\mathsf{Z}^B_{i-1}.\mathtt{hash}, \sigma^B_i\right)$, it holds that:*

$$\Pr_{\mathsf{R}_1,\ldots,\mathsf{R}_i}\left(H^A_i = H^B_i \mid R_1, \ldots, R_{i-1}\right) \leq \frac{1}{T^5}.$$

Proof. From Line 6, we get that the left hand side is the same as:

$$\Pr_{\mathsf{R}_1,\ldots,\mathsf{R}_i}\Big(\mathcal{H}_{\mathsf{sd}^A_i}\left(\mathsf{Z}^A_{i-1}.\mathtt{hash}, \sigma^A_i\right) = \mathcal{H}_{\mathsf{sd}^B_i}\left(\mathsf{Z}^B_{i-1}.\mathtt{hash}, \sigma^B_i\right) \mid R_1, \ldots, R_{i-1}\Big).$$

From Line 5 and Eq. (4.4), we get that this is the same as:

$$\Pr_{\mathsf{R}_1,\ldots,\mathsf{R}_i}\Big(\mathsf{h}_{R^A_i \| R'^A_i}\left(\mathsf{Z}^A_{i-1}.\mathtt{hash}, \sigma^A_i\right) \| R^A_i \| R'^A_i = \mathsf{h}_{R'^B_i \| R^B_i}\left(\mathsf{Z}^B_{i-1}.\mathtt{hash}, \sigma^B_i\right) \| R'^B_i \| R^B_i \mid R_1, \ldots, R_{i-1}\Big).$$

This can be upper bounded by:

$$\Pr_{\mathsf{R}_1,\ldots,\mathsf{R}_i}\Big(\mathsf{h}_{R^A_i \| R^B_i}\left(\mathsf{Z}^A_{i-1}.\mathtt{hash}, \sigma^A_i\right) = \mathsf{h}_{R^A_i \| R^B_i}\left(\mathsf{Z}^B_{i-1}.\mathtt{hash}, \sigma^B_i\right) \mid R_1, \ldots, R_{i-1}\Big).$$

As $\mathsf{R}_1, \ldots, \mathsf{R}_i$ are mutually independent, this is the same as

$$\Pr_{\mathsf{R}_i}\Big(\mathsf{h}_{R^A_i \| R^B_i}\left(\mathsf{Z}^A_{i-1}.\mathtt{hash}, \sigma^A_i\right) = \mathsf{h}_{R^A_i \| R^B_i}\left(\mathsf{Z}^B_{i-1}.\mathtt{hash}, \sigma^B_i\right)\Big).$$

Observe from Line 5 that $\mathsf{R}^A_i, \mathsf{R}^B_i$ are uniformly random. Thus, by Lemma 3.0.1, we can bound this by $\frac{1}{T^{10}}$, and the lemma follows. ∎

LEMMA 5.0.4. *It holds that:*

$$\Pr_{\mathsf{R}_1,\ldots,\mathsf{R}_{M'}}\left(\exists i \in [M'] : \left(\mathsf{Z}^A_{i-1}.\mathtt{hash}, \sigma^A_i\right) \neq \left(\mathsf{Z}^B_{i-1}.\mathtt{hash}, \sigma^B_i\right) \wedge H^A_i = H^B_i\right) \leq \frac{1}{T^3}.$$

Proof. By a union bound and the fact that $M' \leq 2T$ (see Eq. (4.1)), it is enough to show that for all $i \in [M']$, we have:

$$\Pr_{\mathsf{R}_1,\ldots,\mathsf{R}_{M'}}\left(\left(\mathsf{Z}^A_{i-1}.\mathtt{hash}, \sigma^A_i\right) \neq \left(\mathsf{Z}^B_{i-1}.\mathtt{hash}, \sigma^B_i\right) \wedge H^A_i = H^B_i\right) \leq \frac{1}{T^5}.$$

Note that the event inside the $\Pr(\cdot)$ does not depend on $\mathsf{R}_{i+1}, \ldots, \mathsf{R}_{M'}$ and we can remove them from the probability. We shall in fact show a stronger statement that for all $R_1, \ldots, R_{i-1}$ we have:

$$\Pr_{\mathsf{R}_1,\ldots,\mathsf{R}_i}\left(\left(\mathsf{Z}^A_{i-1}.\mathtt{hash}, \sigma^A_i\right) \neq \left(\mathsf{Z}^B_{i-1}.\mathtt{hash}, \sigma^B_i\right) \wedge H^A_i = H^B_i \mid R_1, \ldots, R_{i-1}\right) \leq \frac{1}{T^5}.$$

Now, recall that conditioning on $R_1, \ldots, R_{i-1}$ fixes the value of $\left(\mathsf{Z}^C_{i-1}.\mathtt{hash}, \sigma^C_i\right)$ for $C \in \{A, B\}$. Thus, it is enough to consider $R_1, \ldots, R_{i-1}$ such that $\left(\mathsf{Z}^A_{i-1}.\mathtt{hash}, \sigma^A_i\right) \neq \left(\mathsf{Z}^B_{i-1}.\mathtt{hash}, \sigma^B_i\right)$ and show that:

$$\Pr_{\mathsf{R}_1,\ldots,\mathsf{R}_i}\left(H^A_i = H^B_i \mid R_1, \ldots, R_{i-1}\right) \leq \frac{1}{T^5}.$$

This is exactly Lemma 5.0.3. ∎

LEMMA 5.0.5. *It holds that:*

$$\Pr_{\mathsf{R}_1,\ldots,\mathsf{R}_{M'}}\left(\exists i' < i \in [M'], C \in \{A, B\} : \mathsf{R}^C_i \in \left\{\mathsf{R}^A_{i'}, \mathsf{R}^B_{i'}, \mathsf{R}'^A_{i'}, \mathsf{R}'^B_{i'}\right\}\right) \leq \frac{1}{T^3}.$$

Proof. By a union bound and the fact that $M' \le 2T$ (see Eq. (4.1)), it is enough to show that for all $i' < i \in [M']$ and all $C \in \{A, B\}$, we have:

$$\Pr_{\mathsf{R}_1,\ldots,\mathsf{R}_{M'}}\left(\mathsf{R}_i^C \in \left\{\mathsf{R}_{i'}^A, \mathsf{R}_{i'}^B, \mathsf{R}_{i'}'^A, \mathsf{R}_{i'}'^B\right\}\right) \le \frac{1}{T^8}.$$

Note that the event inside the $\Pr(\cdot)$ does not depend on $\mathsf{R}_{i+1}, \ldots, \mathsf{R}_{M'}$ and we can remove them from the probability. We shall in fact show a stronger statement that for all $R_1, \ldots, R_{i-1}$ we have:

$$\Pr_{\mathsf{R}_1,\ldots,\mathsf{R}_i}\left(\mathsf{R}_i^C \in \left\{\mathsf{R}_{i'}^A, \mathsf{R}_{i'}^B, \mathsf{R}_{i'}'^A, \mathsf{R}_{i'}'^B\right\} \mid R_1, \ldots, R_{i-1}\right) \le \frac{1}{T^8}.$$

Now, observe that conditioning on $R_1, \ldots, R_{i-1}$ fixes the value of $\left\{\mathsf{R}_{i'}^A, \mathsf{R}_{i'}^B, \mathsf{R}_{i'}'^A, \mathsf{R}_{i'}'^B\right\}$. We get from the fact that R_i^C is uniformly random and from Eq. (4.1) that:

$$\Pr_{\mathsf{R}_1,\ldots,\mathsf{R}_i}\left(\mathsf{R}_i^C \in \left\{R_{i'}^A, R_{i'}^B, R_{i'}'^A, R_{i'}'^B\right\} \mid R_1, \ldots, R_{i-1}\right) \le \frac{4}{T^{10}} \le \frac{1}{T^8}.$$

□

For the rest of this proof, we fix an arbitrary $\mathsf{R}_1, \ldots, \mathsf{R}_{M'}$ such that the events in Lemmas 5.0.4 and 5.0.5 do not happen, and show that for any such $\mathsf{R}_1, \ldots, \mathsf{R}_{M'}$, the event in Eq. (5.5) happens. This is enough to show Eq. (5.5).

Observe that as we already fixed the inputs x^A and x^B for Alice and Bob and an adversary Adv, fixing $\mathsf{R}_1, \ldots, \mathsf{R}_{M'}$ fixes the values of all the variables in all the iterations of Algorithm 1. For $C \in \{A, B\}$, we define $\boxed{C9}$ to be the set of all $i \in [M']$ such that Line 9 is executed by party C in iteration i. The notations $\boxed{C12}$, $\boxed{C14}$, $\boxed{C16}$ are defined analogously. Finally, for all $C \in \{A, B\}$ and $i \in \{0\} \cup [M']$, we define $\mathsf{MP}^{\star C}_i = \mathsf{MP}^C_i \cup \{\mathsf{Z}^C_i\}$.

5.2.2 The Functions $\mathsf{ev}(\cdot)$ and $\mathsf{ddl}(\cdot)$ For an integer $l > 0$, define $\mathsf{ev}(l)$ to be the smallest power of 2 that does not divide l and define $\mathsf{ddl}(l) = l + \mathsf{ev}(l)$. Define $\mathsf{ev}(0) = \mathsf{ddl}(0) = \infty$ for convenience.

OBSERVATION 5.0.2. *For all $l > 0$, we have $\mathsf{ev}(\mathsf{ddl}(l)) = \mathsf{ev}(l)$. Furthermore, for all $l, l' \ge 0$, we have:*

$$l \ne l' \implies \mathsf{ddl}(l) \ne \mathsf{ddl}(l').$$

LEMMA 5.0.6. *For all $l \ge 0$ and $z \ge 0$, we have $\mathsf{ddl}(l) \le \mathsf{ddl}(\lfloor l \rfloor_{2^z})$.*

Proof. Let $l' = \lfloor l \rfloor_{2^z}$. If $l' = l$, there is nothing to show, so we assume that $l' < l$. This implies that $\mathsf{ev}(l) \le 2^z \le \frac{1}{2} \cdot \mathsf{ev}(l')$. We get:

$$\mathsf{ddl}(l) \le l + 2^z \le l' + 2^{z+1} \le \mathsf{ddl}(l').$$

□

LEMMA 5.0.7. *Let $l > 0$ and $\lambda < \mathsf{ev}(l)$ be a power of 2. For all $l < l' < l + \lambda$, we have $\mathsf{ddl}(l') \le l + \frac{3}{2} \cdot \lambda$.*

Proof. If $\mathsf{ev}(l') \le \lambda/2$, we are easily done. Otherwise, by definition of $\mathsf{ev}(\cdot)$, we must have $l' = l + \lambda/2$ and our choice of λ implies $\mathsf{ev}(l') = \lambda$. The lemma follows. □

LEMMA 5.0.8. *Consider integers $l > 0$, $l' \le l + \frac{3}{4} \cdot \mathsf{ev}(l)$ such that $\mathsf{ddl}(l') < \mathsf{ddl}(l)$. We have $\mathsf{ddl}(l') \le l + \frac{7}{8} \cdot \mathsf{ev}(l)$.*

Proof. Proof by contradiction. If the lemma is false, we can use the definition of $\mathsf{ddl}(\cdot)$ to get:

$$l + \frac{7}{8} \cdot \mathsf{ev}(l) < \mathsf{ddl}(l') < l + \mathsf{ev}(l).$$

This is impossible if $\mathsf{ev}(l) \le 8$, so we assume otherwise. Observe from the definition of $\mathsf{ev}(l)$ that $\mathsf{ev}(l)/8$ is a power of 2 that divides l. Thus, the previous inequality implies that $\mathsf{ev}(l)/8$ does not divide $\mathsf{ddl}(l')$. Equivalently, we can write $\mathsf{ev}(\mathsf{ddl}(l')) \le \mathsf{ev}(l)/8$. By Observation 5.0.2, we get $\mathsf{ev}(l') \le \mathsf{ev}(l)/8$. This gives a contradiction as:

$$\mathsf{ddl}(l') = l' + \mathsf{ev}(l') \le l + \frac{3}{4} \cdot \mathsf{ev}(l) + \mathsf{ev}(l)/8 = l + \frac{7}{8} \cdot \mathsf{ev}(l).$$

□

LEMMA 5.0.9. *For all $l > 0$, it holds that:*

1. *For all $l < l' < \mathsf{ddl}(l)$, there exists $z \geq 0$ such that $l = \lfloor l' \rfloor_{2^z} - 2^z$.*
2. *For all $l' \geq \mathsf{ddl}(l)$, there does not exist $z \geq 0$ such that $l = \lfloor l' \rfloor_{2^z} - 2^z$.*

Proof. We prove each part in turn:

1. Define z to be such that 2^z is the largest power of 2 that is at most $l' - l$. As we assume that $2^0 = 1 \leq l' - l$, this is well defined. Moreover, as $l' < \mathsf{ddl}(l)$, we have $2^z \leq \mathsf{ev}(l)/2$. Thus, by definition of $\mathsf{ev}(\cdot)$, we have that 2^z divides l. Using this, observe that $\lfloor l' \rfloor_{2^z} = l + 2^z$ finishing the proof.
2. Assume for the sake of contradiction that there exists an $z \geq 0$ such that $l = \lfloor l' \rfloor_{2^z} - 2^z$. This means that 2^z divides l and thus, we must have that $2^z \leq \mathsf{ev}(l)/2$. We get:

$$l = \lfloor l' \rfloor_{2^z} - 2^z > l' - 2^z - 2^z \geq \mathsf{ddl}(l) - \mathsf{ev}(l) = l,$$

a contradiction.

∎

5.2.3 Meeting Points and the Function $\mathsf{Last}(\cdot)$

OBSERVATION 5.0.3. *Let $i \in \{0\} \cup [M']$ and $C \in \{A, B\}$. For all $\mathsf{M} \in \mathsf{MP}_i^C$, we have $\mathsf{M.bn} < \mathsf{Z}_i^C.\mathsf{bn}$. Due to Line 9 and an inductive argument, it follows that the values $\mathsf{M.bn}$ are distinct for all $\mathsf{M} \in \mathsf{MP}^{\star C}_i$.*

LEMMA 5.0.10. *Let $i \in [M']$ and $C \in \{A, B\}$. For all $\mathsf{M} \in \mathsf{MP}_{i-1}^C \setminus \mathsf{MP}^{\star C}_i$, we have $\mathsf{Z}_i^C.\mathsf{bn} = \mathsf{ddl}(\mathsf{M.bn})$ and $i \in \boxed{C9}$. Moreover, if $\mathsf{Z}_{i-1}^C \notin \mathsf{MP}^{\star C}_i$, then we have $\mathsf{Z}_i^C.\mathsf{bn} < \mathsf{Z}_{i-1}^C.\mathsf{bn}$.*

Proof. To start, use Observation 5.0.3 to get that $\mathsf{M.bn} < Z_{i-1}^C.\mathsf{bn}$. We now show that $\mathsf{M.bn} < Z_i^C.\mathsf{bn}$. If $Z_{i-1}^C.\mathsf{bn} \leq Z_i^C.\mathsf{bn}$, this is trivial, if not, we must have $i \in \boxed{C16}$ implying that $\mathsf{M.bn} \leq Z_i^C.\mathsf{bn}$. Due to Observation 5.0.3, this inequality cannot be tight unless $\mathsf{M} = \mathsf{Z}_i^C$ and we are done.

Having shown that $\mathsf{M.bn} < Z_{i-1}^C.\mathsf{bn}, Z_i^C.\mathsf{bn}$, we combine this with $\mathsf{M} \in \mathsf{MP}_{i-1}^C \setminus \mathsf{MP}^{\star C}_i$ and Line 20 and Lemma 5.0.9 to get $\mathsf{Z}_{i-1}^C.\mathsf{bn} < \mathsf{ddl}(\mathsf{M.bn}) \leq \mathsf{Z}_i^C.\mathsf{bn}$. The corollary now follows as $\mathsf{Z.bn}$ increases by at most one in any iteration and increases only when a party executes Line 9. The "moreover" part is simply because $\mathsf{Z}_i^C.\mathsf{bn} \leq \mathsf{Z}_{i-1}^C.\mathsf{bn} + 1$ and Lines 9 and 20. ∎

LEMMA 5.0.11. *Let $0 \leq i' \leq i \leq M'$ and $C \in \{A, B\}$. For all $\mathsf{Z}_{i'}^C.\mathsf{bn} < l \leq \mathsf{Z}_i^C.\mathsf{bn}$, there exists $i' < i'' \leq i$ such that $i'' \in \boxed{C9}$ and $\mathsf{Z}_{i''}^C.\mathsf{bn} = l$.*

Proof. Proof by induction on $i - i'$. The base case $i = i'$ is trivial. We show the result for $i > i'$ assuming it holds for $i-1$. If $l \leq \mathsf{Z}_{i-1}^C.\mathsf{bn}$, the result follows from the induction hypothesis. If not, we must have $\mathsf{Z}_{i-1}^C.\mathsf{bn} < l \leq \mathsf{Z}_i^C.\mathsf{bn}$. Observing Algorithm 1, this only happens if $i \in \boxed{C9}$ and $\mathsf{Z}_i^C.\mathsf{bn} = l$ finishing the proof. ∎

The foregoing lemma (with $i' = 0$) allows us to define:

DEFINITION 5.0.1. *Let $i \in \{0\} \cup [M']$ and $C \in \{A, B\}$. For all $l \in \left[\mathsf{Z}_i^C.\mathsf{bn}\right]$, we define $\mathsf{Last}_i^C(l)$ to be the largest $i' \in [i]$ such that $i' \in \boxed{C9}$ and $\mathsf{Z}_{i'}^C.\mathsf{bn} = l$. We adopt the convention that $\mathsf{Last}_i^C(0) = 0$.*

Our definition satisfies the following properties:

LEMMA 5.0.12. *Let $i \in \{0\} \cup [M']$ and $C \in \{A, B\}$.*

1. *For all $\mathsf{M} \in \mathsf{MP}^{\star C}_i$, we have:*
   - *(a) $\mathsf{M} = \mathsf{Z}^C_{\mathsf{Last}_i^C(\mathsf{M.bn})}$.*
   - *(b) For all $\mathsf{Last}_i^C(\mathsf{M.bn}) \leq i' \leq i$, we have $\mathsf{M} \in \mathsf{MP}^{\star C}_{i'}$ and $\mathsf{Z}_{i'}^C.\mathsf{bn} < \mathsf{ddl}(\mathsf{M.bn})$.*

(c) *For all* $z \geq 0$, *there exists* $\mathsf{M}' \in \mathsf{MP}^{\star C}_i$ *with* $\mathsf{M}'.\mathsf{bn} = \lfloor \mathsf{M}.\mathsf{bn} \rfloor_{2^z}$.

2. *For all* $l \in \left[\mathsf{Z}^C_i.\mathsf{bn}\right]$, *we have:*

(a) *For all* $\mathsf{Last}^C_i(l) \leq i' \leq i$, *it holds that* $\mathsf{Last}^C_{i'}(l) = \mathsf{Last}^C_i(l)$ *and* $\mathsf{Z}^C_{i'}.\mathsf{bn} \geq l$. *As Definition 5.0.1 implies that* $\mathsf{Last}^C_i(l)$ *are distinct for all* l, *it follows that:*

$$0 = \mathsf{Last}^C_i(0) < \mathsf{Last}^C_i(1) < \cdots < \mathsf{Last}^C_i\left(\mathsf{Z}^C_i.\mathsf{bn}\right) \leq i.$$

(b) $\mathsf{Z}^C_{\mathsf{Last}^C_i(l)-1} = \mathsf{Z}^C_{\mathsf{Last}^C_i(l-1)}$.

(c) *If there does not exist* $\mathsf{M} \in \mathsf{MP}^{\star C}_i$ *with* $\mathsf{M}.\mathsf{bn} = l$, *then there exists* $\mathsf{Last}^C_i(l) < i' \leq i$ *such that* $\mathsf{Z}^C_{i'}.\mathsf{bn} \geq \mathsf{ddl}(l)$.

Proof. We prove each part separately.

1. Proof by induction on i. The base case $i = 0$ is straightforward. For Items 1a and 1b, note from Algorithm 1 that either $\mathsf{M} \in \mathsf{MP}^{\star C}_{i-1}$ or $i \in \boxed{C9}$ and $\mathsf{M} = \mathsf{Z}^C_i$, *but not both.* If the latter happens, it is easy to see that $\mathsf{Last}^C_i(\mathsf{M}.\mathsf{bn}) = i$ and therefore Items 1a and 1b hold. We therefore assume that the former happens, and we have from Observation 5.0.3 and Definition 5.0.1 that $\mathsf{Last}^C_i(\mathsf{M}.\mathsf{bn}) = \mathsf{Last}^C_{i-1}(\mathsf{M}.\mathsf{bn})$ and Item 1a follows from the induction hypothesis. For Item 1b, all we have to show is that $\mathsf{Z}^C_i.\mathsf{bn} < \mathsf{ddl}(\mathsf{M}.\mathsf{bn})$. This is because, otherwise, we must have $\mathsf{M} \neq \mathsf{Z}^C_i \in \mathsf{MP}^C_i$ but this contradicts Line 20 and Lemma 5.0.9.

It remains to show Item 1c. Note that this is trivial if $\mathsf{M}.\mathsf{bn} = \lfloor \mathsf{M}.\mathsf{bn} \rfloor_{2^z}$, so we assume that $\mathsf{M}.\mathsf{bn} > \lfloor \mathsf{M}.\mathsf{bn} \rfloor_{2^z}$. We first show that there exists $\mathsf{M}' \in \mathsf{MP}^{\star C}_{i-1}$ with $\mathsf{M}'.\mathsf{bn} = \lfloor \mathsf{M}.\mathsf{bn} \rfloor_{2^z}$. If $\mathsf{M} \in \mathsf{MP}^{\star C}_{i-1}$, this is because of the induction hypothesis. If not, we must have $\mathsf{Z}^C_i.\mathsf{bn} = \mathsf{Z}^C_{i-1}.\mathsf{bn} + 1$ and $\mathsf{M} = \mathsf{Z}^C_i$. Now, $\mathsf{M}.\mathsf{bn} > \lfloor \mathsf{M}.\mathsf{bn} \rfloor_{2^z}$ implies $\lfloor \mathsf{M}.\mathsf{bn} \rfloor_{2^z} = \left\lfloor \mathsf{Z}^C_{i-1}.\mathsf{bn} \right\rfloor_{2^z}$, and we are done by the induction hypothesis.

Next, as either $\mathsf{M} \in \mathsf{MP}^{\star C}_{i-1}$ or not, we have from $\mathsf{M}'.\mathsf{bn} < \mathsf{M}.\mathsf{bn}$ and Observation 5.0.3 that either $\mathsf{M}' \in \mathsf{MP}^C_{i-1}$ or $i \in \boxed{C9}$. Using Lemmas 5.0.6 and 5.0.10, this means that the only way $\mathsf{M}' \notin \mathsf{MP}^{\star C}_i$ is if $\mathsf{Z}^C_i.\mathsf{bn} \geq \mathsf{ddl}(\mathsf{M}'.\mathsf{bn}) \geq \mathsf{ddl}(\mathsf{M}.\mathsf{bn})$. However, this contradicts Item 1b.

2. We have:

(a) If $\mathsf{Last}^C_{i'}(l) \neq \mathsf{Last}^C_i(l)$, we have from Definition 5.0.1 that $i' < \mathsf{Last}^C_i(l)$, a contradiction. If $\mathsf{Z}^C_{i'}.\mathsf{bn} < l$, then the i'' promised by Lemma 5.0.11 contradicts Definition 5.0.1.

(b) Let $i' = \mathsf{Last}^C_i(l)$ for convenience and note that $i' \in \boxed{C9}$ by Definition 5.0.1 implying that $l = \mathsf{Z}^C_{i'}.\mathsf{bn} = \mathsf{Z}^C_{i'-1}.\mathsf{bn} + 1$. We get from Items 1a and 2a that:

$$\mathsf{Z}^C_{i'-1} = \mathsf{Z}^C_{\mathsf{Last}^C_{i'-1}(l-1)} = \mathsf{Z}^C_{\mathsf{Last}^C_i(l-1)}.$$

(c) Proof by induction on i. The base case $i = 0$ is straightforward. For $i > 0$, note that our assumptions imply that $l \leq \mathsf{Z}^C_i.\mathsf{bn} - 1 \leq \mathsf{Z}^C_{i-1}.\mathsf{bn}$. It follows by Definition 5.0.1 that $\mathsf{Last}^C_i(l) = \mathsf{Last}^C_{i-1}(l)$. If there does not exist $\mathsf{M} \in \mathsf{MP}^{\star C}_{i-1}$ with $\mathsf{M}.\mathsf{bn} = l$, we are done by the induction hypothesis. Otherwise, as either $i \in \boxed{C9}$ or $l < \mathsf{Z}^C_{i-1}.\mathsf{bn}$ implying $\mathsf{M} \in \mathsf{MP}^C_{i-1}$, by Lemma 5.0.10, the only reason $\mathsf{M} \notin \mathsf{MP}^{\star C}_i$ is if $\mathsf{Z}^C_i.\mathsf{bn} \geq \mathsf{ddl}(l)$, as desired.

□

5.2.4 The Variable E

Observation 5.0.4. *The variables* E *and* $\mathsf{Z}.\mathsf{bn}$ *are always non-negative and their values never increase by more than* 1 *in one iteration.*

LEMMA 5.0.13. *Let $i \in \{0\} \cup [M']$ and $C \in \{A, B\}$. It holds that:*

$$E_i^C < \mathsf{Z}_i^C.\mathsf{bn} - \max_{\mathsf{M} \in \mathsf{MP}_i^C} \mathsf{M}.\mathsf{bn}.$$

Furthermore, if $i \in \boxed{C16}$, we have $E_i^C = 0$ and:

$$E_{i-1}^C + 1 = \mathsf{Z}_{i-1}^C.\mathsf{bn} - \max_{\mathsf{M} \in \mathsf{MP}_{i-1}^C} \mathsf{M}.\mathsf{bn}.$$

Proof. Proof by induction on i. The base case $i = 0$ is straightforward. We show the result for $i > 0$ assuming it holds for $i - 1$. Consider the following cases:

- **When $i \in \boxed{C9}$:** In this case, we must have $E_{i-1}^C = E_i^C = 0$ and Observation 5.0.3 finishes the proof.
- **When $i \in \boxed{C12}$:** In this case, MP and Z stay unchanged (Line 20 does not affect MP), and we simply have by the induction hypothesis:

$$E_i^C \leq E_{i-1}^C < \mathsf{Z}_{i-1}^C.\mathsf{bn} - \max_{\mathsf{M} \in \mathsf{MP}_{i-1}^C} \mathsf{M}.\mathsf{bn} = \mathsf{Z}_i^C.\mathsf{bn} - \max_{\mathsf{M} \in \mathsf{MP}_i^C} \mathsf{M}.\mathsf{bn}.$$

- **When $i \in \boxed{C14}$ and $i \notin \boxed{C16}$:** In this case too, MP and Z stay unchanged (Line 20 does not affect MP). Moreover, the fact that $i \notin \boxed{C16}$ means that:

$$E_i^C < \mathsf{Z}_i^C.\mathsf{bn} - \max_{\mathsf{M} \in \mathsf{MP}_i^C} \mathsf{M}.\mathsf{bn}.$$

- **When $i \in \boxed{C16}$:** In this case, we have:

$$\mathsf{Z}_{i-1}^C.\mathsf{bn} - \max_{\mathsf{M} \in \mathsf{MP}_{i-1}^C} \mathsf{M}.\mathsf{bn} \leq E_{i-1}^C + 1.$$

  Together with our induction hypothesis, this implies that the inequality must be tight. Observing Line 16, we get that $E_i^C = 0$ and the result follows.

□

5.2.5 The Variable Pre

DEFINITION 5.0.2. *For all $i \in \{0\} \cup [M']$, define the set:*

$$\mathcal{P}_i = \left\{ l \geq 0 \mid \exists \mathsf{M}^A \in \mathsf{MP}^{\star A}_i, \mathsf{M}^B \in \mathsf{MP}^{\star B}_i : \mathsf{M}^A.\mathtt{hash} = \mathsf{M}^B.\mathtt{hash} \wedge \mathsf{M}^A.\mathsf{bn} = \mathsf{M}^B.\mathsf{bn} = l \right\}.$$

Observe that $0 \in \mathcal{P}_i$ for all $i \in \{0\} \cup [M']$. Also, define $\mathsf{Pre}_i = \max \mathcal{P}_i$.

The following follows from Observation 5.0.3.

OBSERVATION 5.0.5. *For all $i \in \{0\} \cup [M']$ and $C \in \{A, B\}$, it holds that $0 \leq \mathsf{Pre}_i \leq \mathsf{Z}_i^C.\mathsf{bn}$.*

LEMMA 5.0.14. *Let $i \in \{0\} \cup [M']$, $l^A \in \left[\mathsf{Z}_i^A.\mathsf{bn}\right]$, $l^B \in \left[\mathsf{Z}_i^B.\mathsf{bn}\right]$ be such that $\mathsf{Z}^A_{\mathsf{Last}_i^A(l^A)}.\mathtt{hash} = \mathsf{Z}^B_{\mathsf{Last}_i^B(l^B)}.\mathtt{hash}$. It holds that:*

$$\left(\mathsf{Last}_i^A(l^A), \sigma^A_{\mathsf{Last}_i^A(l^A)}, \mathsf{Z}^A_{\mathsf{Last}_i^A(l^A-1)}.\mathtt{hash}\right) = \left(\mathsf{Last}_i^B(l^B), \sigma^B_{\mathsf{Last}_i^B(l^B)}, \mathsf{Z}^B_{\mathsf{Last}_i^B(l^B-1)}.\mathtt{hash}\right).$$

Repeatedly applying the above, we also get $l^A = l^B$.

Proof. For convenience, we define $i^C = \mathsf{Last}_i^C\left(l^C\right)$ for all $C \in \{A, B\}$. We first show that $i^A = i^B$ by contradiction. Suppose otherwise and assume without loss of generality that $i^A < i^B$. We have:

$$
\begin{aligned}
&(\text{As } \mathsf{Last}_i^C(l) \in \boxed{C9} \text{ for all } C \in \{A,B\}) \\
&\qquad \mathsf{Z}_{i^A}^A.\mathtt{hash} = \mathsf{Z}_{i^B}^B.\mathtt{hash} \implies H_{i^A}^A = H_{i^B}^B \\
&(\text{Line 6}) \qquad \implies \mathcal{H}_{\mathsf{sd}_{i^A}^A}\left(\mathsf{Z}_{i^A-1}^A.\mathtt{hash}, \sigma_{i^A}^A\right) = \mathcal{H}_{\mathsf{sd}_{i^B}^B}\left(\mathsf{Z}_{i^B-1}^B.\mathtt{hash}, \sigma_{i^B}^B\right) \\
&(\text{Eq. (4.4)}) \qquad \implies \mathsf{sd}_{i^A}^A = \mathsf{sd}_{i^B}^B \\
&(\text{Line 5}) \qquad \implies R'^A_{i^A} = R^B_{i^B},
\end{aligned}
$$

a contradiction to Lemma 5.0.5. Having shown $i^A = i^B$, we call the common value i'. By Definition 5.0.1, we have $i' \in \boxed{A9} \cap \boxed{B9}$ and $\mathsf{Z}_{i'}^C.\mathsf{bn} = l^C$ for all $C \in \{A, B\}$. We get:

$$
\begin{aligned}
&(\text{As } i' \in \boxed{A9} \cap \boxed{B9}) \qquad \mathsf{Z}_{i'}^A.\mathtt{hash} = \mathsf{Z}_{i'}^B.\mathtt{hash} \implies H_{i'}^A = H_{i'}^B \\
&(\text{Lemma 5.0.4}) \qquad \implies \left(\mathsf{Z}_{i'-1}^A.\mathtt{hash}, \sigma_{i'}^A\right) \neq \left(\mathsf{Z}_{i'-1}^B.\mathtt{hash}, \sigma_{i'}^B\right) \\
&(\text{Lemma 5.0.12, Item 2b}) \qquad \implies \left(\mathsf{Z}_{\mathsf{Last}_i^A(l^A-1)}^A.\mathtt{hash}, \sigma_{i'}^A\right) \neq \left(\mathsf{Z}_{\mathsf{Last}_i^B(l^B-1)}^B.\mathtt{hash}, \sigma_{i'}^B\right).
\end{aligned}
$$

Recalling the definition of i', the lemma follows. ∎

COROLLARY 5.0.1. *Let $i \in [M']$ be such that $H_i^A = H_i^B$. It holds that $\mathsf{Pre}_{i-1} = \mathsf{Z}_{i-1}^A.\mathsf{bn} = \mathsf{Z}_{i-1}^B.\mathsf{bn}$.*

Proof. As $H_i^A = H_i^B$, we have from Lemma 5.0.4 that $\mathsf{Z}_{i-1}^A.\mathtt{hash} = \mathsf{Z}_{i-1}^B.\mathtt{hash}$. Now applying Item 1a of Lemma 5.0.12, we get $\mathsf{Z}^A_{\mathsf{Last}_{i-1}^A\left(\mathsf{Z}_{i-1}^A.\mathsf{bn}\right)}.\mathtt{hash} = \mathsf{Z}^B_{\mathsf{Last}_{i-1}^B\left(\mathsf{Z}_{i-1}^B.\mathsf{bn}\right)}.\mathtt{hash}$. From Lemma 5.0.14, we get $\mathsf{Z}_{i-1}^A.\mathsf{bn} = \mathsf{Z}_{i-1}^B.\mathsf{bn}$. Using Definition 5.0.2, observe that this implies $\mathsf{Z}_{i-1}^A.\mathsf{bn} = \mathsf{Z}_{i-1}^B.\mathsf{bn} = \mathsf{Pre}_{i-1}$. ∎

LEMMA 5.0.15. *Let $i \in \{0\} \cup [M']$. The following hold:*

1. *For all $0 \leq l \leq \mathsf{Pre}_i$, we have $\mathsf{Z}^A_{\mathsf{Last}_i^A(l)}.\mathtt{hash} = \mathsf{Z}^B_{\mathsf{Last}_i^B(l)}.\mathtt{hash}$ and $\mathsf{Last}_i^A(l) = \mathsf{Last}_i^B(l)$.*
2. *For all $\mathsf{Last}_i^A(\mathsf{Pre}_i) \leq i' \leq i$, we have $\max\left(\mathsf{Z}_{i'}^A.\mathsf{bn}, \mathsf{Z}_{i'}^B.\mathsf{bn}\right) < \mathsf{ddl}(\mathsf{Pre}_i)$ and $\mathsf{Pre}_i \in \mathcal{P}_{i'}$. It follows that $\mathsf{Pre}_i \leq \mathsf{Pre}_{i'}$.*
3. *For all $0 \leq l \leq \mathsf{Pre}_i$, if there exists $\mathsf{M}^A \in {\mathsf{MP}^\star}_i^A$, $\mathsf{M}^B \in {\mathsf{MP}^\star}_i^B$ such that $\mathsf{M}^A.\mathsf{bn} = \mathsf{M}^B.\mathsf{bn} = l$, then $\mathsf{M}^A.\mathtt{hash} = \mathsf{M}^B.\mathtt{hash}$. It follows that $l \in \mathcal{P}_i$.*

Proof. We prove each part separately:

1. The second part follows from the first due to Lemma 5.0.14. For the first, notice from Lemma 5.0.14 that it is sufficient to show the result for $l = \mathsf{Pre}_i$. For this, we use Definition 5.0.2 to get $\mathsf{M}^A \in {\mathsf{MP}^\star}_i^A$ and $\mathsf{M}^B \in {\mathsf{MP}^\star}_i^B$ such that $\mathsf{M}^A.\mathtt{hash} = \mathsf{M}^B.\mathtt{hash}$ and $\mathsf{M}^A.\mathsf{bn} = \mathsf{M}^B.\mathsf{bn} = \mathsf{Pre}_i$. The result now follows from Item 1a of Lemma 5.0.12.
2. By Definition 5.0.2, there exists $\mathsf{M}^A \in {\mathsf{MP}^\star}_i^A$ and $\mathsf{M}^B \in {\mathsf{MP}^\star}_i^B$ such that $\mathsf{M}^A.\mathtt{hash} = \mathsf{M}^B.\mathtt{hash}$ and $\mathsf{M}^A.\mathsf{bn} = \mathsf{M}^B.\mathsf{bn} = \mathsf{Pre}_i$. Using Item 1b of Lemma 5.0.12, the result follows.
3. By Item 1a of Lemma 5.0.12, we have $\mathsf{M}^A = \mathsf{Z}^A_{\mathsf{Last}_i^A(l)}$ and $\mathsf{M}^B = \mathsf{Z}^B_{\mathsf{Last}_i^B(l)}$. The result follows from Item 1.

∎

LEMMA 5.0.16. *Let $i \in [M']$ be such that $\mathsf{Pre}_{i-1} < \mathsf{Pre}_i$. We have $i \in \boxed{A9} \cap \boxed{B9}$.*

Proof. Recall that $\mathsf{Pre}_i \in \mathcal{P}_i$ and let M^A and M^B be as in Definition 5.0.2. If $\mathsf{Pre}_{i-1} < \mathsf{Pre}_i$, there exists $C \in \{A, B\}$ such that $\mathsf{M}^C \notin {\mathsf{MP}^\star}_{i-1}^C$. As $\mathsf{M}^C \in {\mathsf{MP}^\star}_i^C$ this is only possible if $i \in \boxed{C9}$ and $\mathsf{M}^C = \mathsf{Z}_i^C$. Conclude from this and Definition 5.0.1 that $\mathsf{Last}_i^C(\mathsf{Pre}_i) = i$. Now use Item 1 of Lemma 5.0.15 to get that $\mathsf{Last}_i^A(\mathsf{Pre}_i) = \mathsf{Last}_i^B(\mathsf{Pre}_i) = i$ implying that $i \in \boxed{A9} \cap \boxed{B9}$. ∎

LEMMA 5.0.17. *Let $i \in [M']$ be such that $\mathsf{Pre}_i < \mathsf{Pre}_{i-1}$. It holds that:*

$$\mathcal{P}_i = \Big\{0 \leq l \leq \mathsf{Pre}_{i-1} \mid \exists \mathsf{M}^A \in \mathsf{MP}^{\star A}_i, \mathsf{M}^B \in \mathsf{MP}^{\star B}_i : \mathsf{M}^A.\mathsf{bn} = \mathsf{M}^B.\mathsf{bn} = l\Big\}.$$

Proof. We let S denote the set on the right. Note that $\mathcal{P}_i \subseteq S$ is straightforward from Definition 5.0.2 as all elements in $\mathcal{P}_i$ are at most $\mathsf{Pre}_i < \mathsf{Pre}_{i-1}$. We now show that $S \subseteq \mathcal{P}_i$ by fixing an arbitrary $l \in S$ and showing that $l \in \mathsf{Pre}_i$. As $l \in S$, we have $\mathsf{M}^A \in \mathsf{MP}^{\star A}_i$ and $\mathsf{M}^B \in \mathsf{MP}^{\star B}_i$ such that $\mathsf{M}^A.\mathsf{bn} = \mathsf{M}^B.\mathsf{bn} = l$. Thus, it suffices to show that $\mathsf{M}^A.\mathtt{hash} = \mathsf{M}^B.\mathtt{hash}$. For this, we first claim that for all $C \in \{A, B\}$, we have $\mathsf{M}^C \in \mathsf{MP}^{\star C}_{i-1}$. Indeed, if this is not true for some $C \in \{A, B\}$, then we have from Algorithm 1 that $\mathsf{M}^C.\mathsf{bn} = l = \mathsf{Z}^C_{i-1}.\mathsf{bn} + 1$, which is impossible as $l \leq \mathsf{Pre}_{i-1}$. Together with this claim, Item 3 of Lemma 5.0.15 says that $\mathsf{M}^A.\mathtt{hash} = \mathsf{M}^B.\mathtt{hash}$, as needed. ∎

COROLLARY 5.0.2. *Let $i \in [M']$ be such that $\mathsf{Pre}_i < \mathsf{Pre}_{i-1}$. It holds that:*

$$\lfloor \mathsf{Pre}_{i-1} \rfloor_{\mathsf{ev}(\mathsf{Pre}_{i-1})} \leq \mathsf{Pre}_i.$$

Proof. Define $l = \lfloor \mathsf{Pre}_{i-1} \rfloor_{\mathsf{ev}(\mathsf{Pre}_{i-1})}$ for convenience and observe that $l < \mathsf{Pre}_{i-1}$. Use Definition 5.0.2 and Item 1c of Lemma 5.0.12 to get that there exists $\mathsf{M}'^A \in \mathsf{MP}^{\star A}_{i-1}$ and $\mathsf{M}'^B \in \mathsf{MP}^{\star B}_{i-1}$ such that $\mathsf{M}'^A.\mathsf{bn} = \mathsf{M}'^B.\mathsf{bn} = l$. As $l < \mathsf{Pre}_{i-1}$, we can conclude that $\mathsf{M}'^C \in \mathsf{MP}^C_{i-1}$ for all $C \in \{A, B\}$. We now claim that $\mathsf{M}'^C \in \mathsf{MP}^{\star C}_i$ for all $C \in \{A, B\}$ as if not, we have by Lemmas 5.0.6 and 5.0.10 that there exists $C \in \{A, B\}$ satisfying $\mathsf{Z}^C_i.\mathsf{bn} \geq \mathsf{ddl}(l) > \mathsf{ddl}(\mathsf{Pre}_{i-1})$. As $\mathsf{ddl}(\mathsf{Pre}_{i-1}) > \mathsf{Z}^C_{i-1}.\mathsf{bn}$ by Item 2 of Lemma 5.0.15, this contradicts Observation 5.0.4. Together with this claim, Lemma 5.0.17 implies $l \in \mathcal{P}_i$, and the result follows. ∎

5.2.6 The Variable K For $i \in \{0\} \cup [M']$ and $C \in \{A, B\}$, we define:

$$\mathsf{K}^C_i = 1 - 6 \cdot \mathbb{1}\big(\mathsf{Pre}_i = \mathsf{Z}^C_i.\mathsf{bn}\big). \tag{5.6}$$

LEMMA 5.0.18. *Let $i \in \{0\} \cup [M']$ and $C \in \{A, B\}$. It holds that:*

$$\mathsf{K}^C_i \cdot E^C_i \leq \mathsf{Z}^C_i.\mathsf{bn} - \mathsf{Pre}_i.$$

Proof. If $\mathsf{K}^C_i < 0$, then we are done by Observations 5.0.4 and 5.0.5. If $\mathsf{K}^C_i \geq 0$, we have by Eq. (5.6) that $\mathsf{K}^C_i = 1$ and $\mathsf{Pre}_i \neq \mathsf{Z}^C_i.\mathsf{bn}$. By Definition 5.0.2, this can only happen if $\mathsf{Pre}_i \leq \max_{\mathsf{M} \in \mathsf{MP}^C_i} \mathsf{M}.\mathsf{bn}$. We get by Lemma 5.0.13 that:

$$\mathsf{K}^C_i \cdot E^C_i = E^C_i \leq \mathsf{Z}^C_i.\mathsf{bn} - \max_{\mathsf{M} \in \mathsf{MP}^C_i} \mathsf{M}.\mathsf{bn} \leq \mathsf{Z}^C_i.\mathsf{bn} - \mathsf{Pre}_i.$$

∎

LEMMA 5.0.19. *Let $i \in [M']$ and $C \in \{A, B\}$. It holds that:*

$$\mathsf{K}^C_{i-1} \cdot E^C_{i-1} - \mathsf{Z}^C_{i-1}.\mathsf{bn} \leq 6 + \mathsf{K}^C_i \cdot E^C_i - \mathsf{Z}^C_i.\mathsf{bn}.$$

Proof. Observe from Eq. (5.6) that K is never 0. We consider the following cases:

- **When $\mathsf{K}^C_{i-1} < 0$:** In this case, we have:

$$\begin{aligned} \mathsf{K}^C_{i-1} \cdot E^C_{i-1} - \mathsf{Z}^C_{i-1}.\mathsf{bn} &= -5 \cdot E^C_{i-1} - \mathsf{Z}^C_{i-1}.\mathsf{bn} \\ \text{(Observation 5.0.4)} \quad &\leq 6 - 5 \cdot E^C_i - \mathsf{Z}^C_i.\mathsf{bn} \\ \text{(Observation 5.0.4)} \quad &\leq 6 + \mathsf{K}^C_i \cdot E^C_i - \mathsf{Z}^C_i.\mathsf{bn}. \end{aligned}$$

- **When $\mathsf{K}^C_{i-1}, \mathsf{K}^C_i > 0$:** This means $\mathsf{K}^C_{i-1} = \mathsf{K}^C_i = 1$ and these factors can be ignored. Consider the following subcases:
  - **When $E^C_{i-1} \leq E^C_i + 5$:** In this case, we simply note from Observation 5.0.4 that:

$$E^C_{i-1} - \mathsf{Z}^C_{i-1}.\mathsf{bn} \leq 5 + E^C_i - \mathsf{Z}^C_{i-1}.\mathsf{bn} \leq 6 + E^C_i - \mathsf{Z}^C_i.\mathsf{bn}.$$

– **When $E_i^C + 5 < E_{i-1}^C$:** Observe from Algorithm 1 that this only happens when $i \in \boxed{C16}$. Applying Lemma 5.0.13, we get that $E_i^C = 0$ and we have:

$$E_{i-1}^C - \mathsf{Z}_{i-1}^C.\mathsf{bn} = -\max_{\mathsf{M} \in \mathsf{MP}_{i-1}^C} \mathsf{M.bn} - 1 \leq -\mathsf{Z}_{i-1}^C.\mathsf{bn} \leq 6 + E_i^C - \mathsf{Z}_i^C.\mathsf{bn}.$$

- **When $\mathsf{K}_i^C < 0 < \mathsf{K}_{i-1}^C$:** This means $\mathsf{K}_{i-1}^C = 1$ and this factor can be ignored. We have by Eq. (5.6) and Observation 5.0.5 that $\mathsf{Pre}_i = \mathsf{Z}_i^C.\mathsf{bn}$ and $\mathsf{Pre}_{i-1} < \mathsf{Z}_{i-1}^C.\mathsf{bn}$. We consider the following subcases:
  – **When $\mathsf{Z}_{i-1}^C.\mathsf{bn} \leq \mathsf{Z}_i^C.\mathsf{bn}$:** This means that $\mathsf{Pre}_{i-1} < \mathsf{Pre}_i$. We can apply Lemma 5.0.16 to get $i \in \boxed{A9} \cap \boxed{B9}$. We get $E_{i-1}^C = E_i^C = 0$ and the lemma is trivial.
  – **When $\mathsf{Z}_i^C.\mathsf{bn} < \mathsf{Z}_{i-1}^C.\mathsf{bn}$:** Observe from Algorithm 1 that this only happens when $i \in \boxed{C16}$. Applying Lemma 5.0.13, we get that $E_i^C = 0$ and we have:

$$E_{i-1}^C - \mathsf{Z}_{i-1}^C.\mathsf{bn} = -\max_{\mathsf{M} \in \mathsf{MP}_{i-1}^C} \mathsf{M.bn} - 1 \leq -\mathsf{Z}_{i-1}^C.\mathsf{bn} \leq 6 + \mathsf{K}_i^C \cdot E_i^C - \mathsf{Z}_i^C.\mathsf{bn}.$$

□

5.2.7 Corruptions Recall Eq. (4.1). For $i \in [M']$, we define Err_i to be the indicator variable that is 1 if and only if the number of corruptions made by the adversary in iteration i is at least $\epsilon' B^* > r'/10$. For a subset $I \subseteq [M']$, we define $\mathsf{Err}_I = \sum_{i \in I} \mathsf{Err}_i$. When $I = [i]$ for some $i \in [M']$, we may instead write $\mathsf{Err}_{\leq i}$. We define $\mathsf{Err}_{<i}$ analogously.

LEMMA 5.0.20. *Let $i \in [M']$ be such that $\mathsf{Err}_i = 0$. The following hold:*

1. *We have:*
$$\left(R_i^A, H_i^A, E_{i-1}^A, b_i^A\right) = \left(R_i'^B, H_i'^B, E_{i-1}'^B, b_i'^B\right),$$
and likewise with A and B reversed.
2. *We have $i \in \boxed{A9} \iff i \in \boxed{B9}$ and $i \in \boxed{A12} \iff i \in \boxed{B12}$.*
3. *We have $\sigma_i^A = \sigma_i^B$.*
4. *If $i \notin \boxed{A9} \cup \boxed{A12}$, we have $\mathsf{Z}_{i-1}^A.\mathtt{hash} \neq \mathsf{Z}_{i-1}^B.\mathtt{hash}$.*
5. *If $i \in \boxed{A9}$, we have $\mathsf{Pre}_{i-1} + 1 = \mathsf{Pre}_i = \mathsf{Z}_i^A.\mathsf{bn} = \mathsf{Z}_i^B.\mathsf{bn}$.*
6. *If $i \notin \boxed{A9}$, we have $\mathsf{Pre}_{i-1} = \mathsf{Pre}_i$ and:*
$$\sum_{C \in \{A,B\}} \left(\mathsf{K}_{i-1}^C \cdot E_{i-1}^C - \mathsf{Z}_{i-1}^C.\mathsf{bn}\right) < \sum_{C \in \{A,B\}} \left(\mathsf{K}_i^C \cdot E_i^C - \mathsf{Z}_i^C.\mathsf{bn}\right).$$

Proof. We prove each part separately.

1. Recall that Lines 5 and 7 implicitly use the code ECC that outputs encodings of length $1000r'$. As $\mathsf{Err}_i = 0$, at most $r'/10$ bits of this encoding will be corrupted implying that the parties will decode each others messages correctly. The result follows. We remark that it is E_{i-1} instead of E_i as the value of E may change in iteration i.
2. Follows from the previous part and Line 8.
3. Recall that the parties execute a protocol satisfying Theorem 3.1 in Line 3. Moreover, the fact that $\mathsf{Err}_i = 0$ implies there are at most $\epsilon' B^*$ corruptions in this execution. We get from Theorem 3.1 that the output is the same as the output of a noiseless execution, which satisfies $\sigma_i^A = \sigma_i^B$.

4. Note from the fact that $i \notin \boxed{A9} \cup \boxed{A12}$ and Item 1 that $H_i^A \neq H_i^B$. By Line 6, we get that $\mathcal{H}_{\mathsf{sd}_i^A}\big(\mathsf{Z}_{i-1}^A.\mathtt{hash}, \sigma_i^A\big) \neq \mathcal{H}_{\mathsf{sd}_i^B}\big(\mathsf{Z}_{i-1}^B.\mathtt{hash}, \sigma_i^B\big)$. Now, observe that Item 1 implies $\mathsf{sd}_i^A = \mathsf{sd}_i^B$. Using sd to denote the common value, we get from Eq. (4.4) that $\mathsf{h}_{\mathsf{sd}}\big(\mathsf{Z}_{i-1}^A.\mathtt{hash}, \sigma_i^A\big) \neq \mathsf{h}_{\mathsf{sd}}\big(\mathsf{Z}_{i-1}^B.\mathtt{hash}, \sigma_i^B\big)$. This implies that $\big(\mathsf{Z}_{i-1}^A.\mathtt{hash}, \sigma_i^A\big) \neq \big(\mathsf{Z}_{i-1}^B.\mathtt{hash}, \sigma_i^B\big)$ and Item 3 finishes the proof.

5. As $i \in \boxed{A9}$, we have from Items 1 and 2 that $i \in \boxed{A9} \cap \boxed{B9}$ and $H_i^A = H_i^B$. Using Corollary 5.0.1, we get $\mathsf{Pre}_{i-1} = \mathsf{Z}_{i-1}^A.\mathsf{bn} = \mathsf{Z}_{i-1}^B.\mathsf{bn}$ and all that remains to show is $\mathsf{Pre}_i = \mathsf{Z}_i^A.\mathsf{bn} = \mathsf{Z}_i^B.\mathsf{bn}$. For this, simply note that $H_i^A = H_i^B$ with $i \in \boxed{A9} \cap \boxed{B9}$ implies $\mathsf{Z}_i^A.\mathtt{hash} = \mathsf{Z}_i^B.\mathtt{hash}$ and apply Definition 5.0.2.

6. Due to Lemma 5.0.16, it suffices to show that $\mathsf{Pre}_{i-1} \leq \mathsf{Pre}_i$. We show this by contradiction. If not, recall that $\mathsf{Pre}_{i-1} \in \mathcal{P}_{i-1}$ and let $\mathsf{M}^A \in {\mathsf{MP}^\star}_{i-1}^A$, $\mathsf{M}^B \in {\mathsf{MP}^\star}_{i-1}^B$ be as in Definition 5.0.2. Observe that $\mathsf{Pre}_i < \mathsf{Pre}_{i-1}$ can happen only if there exists $C \in \{A, B\}$ such that $\mathsf{M}^C \notin {\mathsf{MP}^\star}_i$. Using Item 2 and Lemma 5.0.10, this happens only if $i \in \boxed{C16}$ and $\mathsf{M}^C = \mathsf{Z}_{i-1}^C$ implying $\mathsf{M}^C.\mathsf{bn} = \mathsf{Z}_{i-1}^C.\mathsf{bn} = \mathsf{Pre}_{i-1}$. Now, using Item 4, we get that $\mathsf{Z}_{i-1}^A.\mathtt{hash} \neq \mathsf{Z}_{i-1}^B.\mathtt{hash}$. Due to Observation 5.0.3 and Definition 5.0.2, this also means that $\mathsf{Z}_{i-1}^A.\mathsf{bn} \neq \mathsf{Z}_{i-1}^B.\mathsf{bn}$. Assume $C = A$ without loss of generality. Using Observation 5.0.5, we actually get $\mathsf{Pre}_{i-1} = \mathsf{Z}_{i-1}^A.\mathsf{bn} < \mathsf{Z}_{i-1}^B.\mathsf{bn}$. This means that $b_i^A < b_i^B = b_i'^A$ by Item 1. This contradicts $i \in \boxed{A16}$ and we are done.

   We now show the remainder of Item 6. For this, recall Item 2 and consider the following cases:

   - **When** $i \in \boxed{A12} \cap \boxed{B12}$**:** In this case, the values of ${\mathsf{MP}^\star}^A$, ${\mathsf{MP}^\star}^B$, and therefore also of Pre remain unchanged in iteration i. Moreover, we have $E_i^A \leq E_{i-1}^A$ and $E_i^B \leq E_{i-1}^B$. We use Item 1 to get that $H_i^A = H_i^B$ and that there exists $C \in \{A, B\}$ such that $E_{i-1}^C > 0 \implies E_{i-1}^C > E_i^C$. It follows that $E_i^A + E_i^B < E_{i-1}^A + E_{i-1}^B$. Using Corollary 5.0.1, we get $\mathsf{Pre}_{i-1} = \mathsf{Z}_{i-1}^A.\mathsf{bn} = \mathsf{Z}_{i-1}^B.\mathsf{bn} = \mathsf{Pre}_i = \mathsf{Z}_i^A.\mathsf{bn} = \mathsf{Z}_i^B.\mathsf{bn}$. We get:

   $$\begin{aligned} &\text{(Eq. (5.6))} && \sum_{C\in\{A,B\}} \big(\mathsf{K}_{i-1}^C \cdot E_{i-1}^C - \mathsf{Z}_{i-1}^C.\mathsf{bn}\big) \leq \sum_{C\in\{A,B\}} \big(\mathsf{K}_i^C \cdot E_{i-1}^C - \mathsf{Z}_i^C.\mathsf{bn}\big) \\ &\text{(As } \mathsf{K}_i^A = \mathsf{K}_i^B < 0 \text{ and } E_i^A + E_i^B < E_{i-1}^A + E_{i-1}^B\text{)} && < \sum_{C\in\{A,B\}} \big(\mathsf{K}_i^C \cdot E_i^C - \mathsf{Z}_i^C.\mathsf{bn}\big). \end{aligned}$$

   - **When** $i \notin \boxed{A12} \cup \boxed{B12}$**:** In this case, we claim that for all $C \in \{A, B\}$, we have

   $$\mathsf{K}_{i-1}^C \cdot E_{i-1}^C - \mathsf{Z}_{i-1}^C.\mathsf{bn} + \mathbb{1}\big(i \in \boxed{C14}\big) = \mathsf{K}_i^C \cdot E_i^C - \mathsf{Z}_i^C.\mathsf{bn}.$$

   The result easily follows from Line 13 and Item 1 and thus it suffices to show the claim. Fix $C \in \{A, B\}$. If $i \notin \boxed{C14}$, the claim trivially holds due to Eq. (5.6) and the fact that $\mathsf{Pre}_i = \mathsf{Pre}_{i-1}$. Thus, we assume that $i \in \boxed{C14}$.

   We first show that $\mathsf{Pre}_{i-1} < \mathsf{Z}_{i-1}^C.\mathsf{bn}$ implying by Eq. (5.6) that $\mathsf{K}_{i-1}^C = 1$. We show this by contradiction. Suppose not. Then, we have from Observation 5.0.5 that $\mathsf{Pre}_{i-1} = \mathsf{Z}_{i-1}^C.\mathsf{bn}$. We also have from Item 4 that $\mathsf{Z}_{i-1}^A.\mathtt{hash} \neq \mathsf{Z}_{i-1}^B.\mathtt{hash}$. Let $\overline{C}$ be the unique element in $\{A, B\}$ that is different from C. By Definition 5.0.2 and Observation 5.0.3 these two are only possible if $\mathsf{Z}_{i-1}^A.\mathsf{bn} \neq \mathsf{Z}_{i-1}^B.\mathsf{bn}$ which by Observation 5.0.5 implies that $\mathsf{Z}_{i-1}^C.\mathsf{bn} < \mathsf{Z}_{i-1}^{\overline{C}}.\mathsf{bn}$. However, due to Item 1, this means that $b_i^C < b_i'^C$, a contradiction to $i \in \boxed{C14}$.

   Having shown, $\mathsf{K}_{i-1}^C = 1$, if $i \notin \boxed{C16}$, we get $\mathsf{Z}_i^C.\mathsf{bn} = \mathsf{Z}_{i-1}^C.\mathsf{bn}$ implying (as $\mathsf{Pre}_i = \mathsf{Pre}_{i-1}$) that $\mathsf{K}_i^C = \mathsf{K}_{i-1}^C = 1$. We get:

   $$\mathsf{K}_{i-1}^C \cdot E_{i-1}^C - \mathsf{Z}_{i-1}^C.\mathsf{bn} + 1 = E_{i-1}^C - \mathsf{Z}_{i-1}^C.\mathsf{bn} + 1 = E_i^C - \mathsf{Z}_i^C.\mathsf{bn} = \mathsf{K}_i^C \cdot E_i^C - \mathsf{Z}_i^C.\mathsf{bn}.$$

   On the other hand, if $i \notin \boxed{C16}$, we have by Lemma 5.0.13 that $E_i^C = 0$:

   $$\mathsf{K}_{i-1}^C \cdot E_{i-1}^C - \mathsf{Z}_{i-1}^C.\mathsf{bn} + 1 = -\max_{\mathsf{M}\in\mathsf{MP}_{i-1}^C} \mathsf{M}.\mathsf{bn} = -\mathsf{Z}_i^C.\mathsf{bn} = \mathsf{K}_i^C \cdot E_i^C - \mathsf{Z}_i^C.\mathsf{bn},$$

   where the penultimate inequality is because of Line 17 and the fact that $\mathsf{Z}_{i-1}^C.\mathsf{bn} > \mathsf{Pre}_{i-1} \geq 0$ implying $\mathsf{MP}_{i-1}^C \neq \emptyset$.

∎

COROLLARY 5.0.3. *Let $i \in [M']$ be such that $\mathsf{Err}_i = 0$. It holds that:*

$$11 \cdot \mathsf{Pre}_{i-1} + \sum_{C \in \{A,B\}} 5 \cdot \left(\mathsf{K}^C_{i-1} \cdot E^C_{i-1} - \mathsf{Z}^C_{i-1}.\mathsf{bn}\right) < 11 \cdot \mathsf{Pre}_i + \sum_{C \in \{A,B\}} 5 \cdot \left(\mathsf{K}^C_i \cdot E^C_i - \mathsf{Z}^C_i.\mathsf{bn}\right).$$

Proof. From Item 2 of Lemma 5.0.20, we have $i \in \boxed{A9} \iff i \in \boxed{B9}$. If both of these are true, we observe from Line 9 that the terms corresponding to E are 0 and all the other quantities increase by exactly 1 (see Line 9 and Item 5 of Lemma 5.0.20). On the other hand, if they are both false, we simply use Item 6 of Lemma 5.0.20. ∎

5.2.8 Taxes Important to our analysis is the following notion of a "tax" associated with each "correct" meeting point. Let $i \in \{0\} \cup [M']$. Define the sets:

$$\mathsf{CMP}_i = \bigcup_{C \in \{A,B\}} \mathsf{CMP}^C_i \quad \text{where} \quad \mathsf{CMP}^C_i = \left\{0 \le l \le \mathsf{Pre}_i \mid \exists \mathsf{M} \in \mathsf{MP}^{\star C}_i : \mathsf{M.bn} = l\right\} \tag{5.7}$$

$$\mathsf{S}_i = \{\mathsf{Pre}_i\} \cup \left\{\max_{\mathsf{M} \in \mathsf{MP}^C_i} \mathsf{M.bn}\right\}_{C \in \{A,B\}} \cup \left\{\arg\min_{\mathsf{M} \in \mathsf{CMP}^C_i} \mathsf{ddl}(\mathsf{M.bn})\right\}_{C \in \{A,B\}}. \tag{5.8}$$

We next define, for $i \in \{0\} \cup [M']$ functions $\mathsf{Tax}_i : \mathsf{CMP}_i \to \mathbb{Z}$ inductively as follows: For $i = 0$, observe that $\mathsf{CMP}_i = \{0\}$ and define $\mathsf{Tax}_i(0) = 0$. For $i > 0$ and $l \in \mathsf{CMP}_i$, define:

$$\mathsf{Tax}_i(l) = \begin{cases} 0, & \text{if } l \notin \mathsf{CMP}_{i-1} \\ \mathsf{Tax}_{i-1}(l) + \mathbb{1}(\mathsf{Err}_i > 0 \wedge l \in \mathsf{S}_i), & \text{if } l \in \mathsf{CMP}_{i-1} \end{cases}. \tag{5.9}$$

LEMMA 5.0.21. *Let $i \in \{0\} \cup [M']$ and $C \in \{A, B\}$. Let $0 \le l \le \mathsf{Pre}_i$ be such that there exists $\mathsf{M} \in \mathsf{MP}^{\star C}_i$ with $\mathsf{M.bn} = l$. For all $\mathsf{Last}^A_i(\mathsf{Pre}_i) \le i' \le i$, we have $l \in \mathsf{CMP}^C_{i'} \subseteq \mathsf{CMP}_{i'}$.*

Proof. Fix i, C, l, i' as in the lemma statement. By Item 2 of Lemma 5.0.15, we have $\mathsf{Pre}_i \le \mathsf{Pre}_{i'}$. It is thus enough to show that $\mathsf{M} \in \mathsf{MP}^{\star C}_{i'}$, where M is as promised by the lemma. This is because of the fact that $\mathsf{Last}^A_i(\mathsf{Pre}_i) = \mathsf{Last}^B_i(\mathsf{Pre}_i)$ (which is due to Item 1 of Lemma 5.0.15) and Item 1b of Lemma 5.0.12. ∎

Using Eq. (5.9) repeatedly, we also get:

COROLLARY 5.0.4. *Let $i \in \{0\} \cup [M']$, $0 \le l \le \mathsf{Pre}_i$ be such that there exists $\mathsf{M} \in \mathsf{MP}^{\star A}_i \cup \mathsf{MP}^{\star B}_i$ with $\mathsf{M.bn} = l$. It holds that:*

$$\mathsf{Tax}_i(l) \ge \sum_{\mathsf{Last}^A_i(\mathsf{Pre}_i) < i' \le i} \mathbb{1}(\mathsf{Err}_{i'} > 0 \wedge l \in \mathsf{S}_{i'}).$$

5.2.9 Understanding Taxes

LEMMA 5.0.22. *Let $i \in [M']$. We have:*

$$\sum_{l \in \mathsf{CMP}_i} \mathsf{Tax}_i(l) \le 5 \cdot \mathbb{1}(\mathsf{Err}_i > 0) + \sum_{l \in \mathsf{CMP}_i \cap \mathsf{CMP}_{i-1}} \mathsf{Tax}_{i-1}(l).$$

Proof. We have:

$$\begin{aligned}
\text{(Eq. (5.9))} \quad \sum_{l \in \mathsf{CMP}_i} \mathsf{Tax}_i(l) &\le \sum_{l \in \mathsf{CMP}_i \cap \mathsf{CMP}_{i-1}} \mathsf{Tax}_i(l) \\
\text{(Eq. (5.9))} \quad &\le \sum_{l \in \mathsf{CMP}_i \cap \mathsf{CMP}_{i-1}} \mathsf{Tax}_{i-1}(l) + \sum_{l \in \mathsf{CMP}_i \cap \mathsf{CMP}_{i-1}} \mathbb{1}(\mathsf{Err}_i > 0 \wedge l \in \mathsf{S}_i) \\
&\le \sum_{l \in \mathsf{CMP}_i \cap \mathsf{CMP}_{i-1}} \mathsf{Tax}_{i-1}(l) + \mathbb{1}(\mathsf{Err}_i > 0) \cdot |\mathsf{S}_i| \\
\text{(Eq. (5.8))} \quad &\le \sum_{l \in \mathsf{CMP}_i \cap \mathsf{CMP}_{i-1}} \mathsf{Tax}_{i-1}(l) + 5 \cdot \mathbb{1}(\mathsf{Err}_i > 0).
\end{aligned}$$

∎

LEMMA 5.0.23. *Let $C \in \{A, B\}$ and $i \in \boxed{C9}$ be such that $\mathsf{Z}_i^C.\mathsf{bn} = \mathsf{ddl}(\mathsf{Pre}_{i-1})$. It holds that:*

$$\mathsf{Pre}_{i-1} - \mathsf{Pre}_i \leq 8 \cdot \mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}) + 8.$$

Proof. If $\mathsf{Pre}_{i-1} - \mathsf{Pre}_i \leq 8$, this is trivial, so we assume otherwise. By Corollary 5.0.2, we get $8 < \mathsf{Pre}_{i-1} - \mathsf{Pre}_i \leq \mathsf{Pre}_{i-1} - \lfloor \mathsf{Pre}_{i-1} \rfloor_{\mathsf{ev}(\mathsf{Pre}_{i-1})} = \frac{1}{2} \cdot \mathsf{ev}(\mathsf{Pre}_{i-1})$. It follows that $\mathsf{ev}(\mathsf{Pre}_{i-1}) \geq 32$. Moreover, from Corollary 5.0.4 and the fact that $\mathsf{Last}_{i-1}^A(\mathsf{Pre}_{i-1}) = \mathsf{Last}_{i-1}^C(\mathsf{Pre}_{i-1})$ (see Item 1 of Lemma 5.0.15), we get:

$$\mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}) \geq \sum_{\mathsf{Last}_{i-1}^C(\mathsf{Pre}_{i-1}) < i' < i} \mathbb{1}(\mathsf{Err}_{i'} > 0 \wedge \mathsf{Pre}_{i-1} \in \mathsf{S}_{i'}). \tag{5.10}$$

In the remainder of this proof we simply show that the right hand side above is at least $\frac{1}{16} \cdot \mathsf{ev}(\mathsf{Pre}_{i-1}) - 1$. For this, define:

$$l_1 = \mathsf{Pre}_{i-1} \qquad l_2 = \frac{l_1 + l_5}{2} \qquad l_3 = \frac{l_1 + 3l_5}{4} \qquad l_4 = \frac{l_1 + 7l_5}{8} \qquad l_5 = \mathsf{ddl}(\mathsf{Pre}_{i-1}). \tag{5.11}$$

As $\mathsf{ev}(\mathsf{Pre}_{i-1}) \geq 32$, we get that all of these are integers and $l_1 < l_2 < l_3 < l_4 < l_5$. Observe that $i \in \boxed{C9}$ implies $\mathsf{Z}_{i-1}^C.\mathsf{bn} = l_5 - 1$.We consider the following cases:

- **When $\mathsf{Z}^A_{\mathsf{Last}^C_{i-1}(l_3)}.\mathtt{hash} \neq \mathsf{Z}^B_{\mathsf{Last}^C_{i-1}(l_3)}.\mathtt{hash}$:** In this case, Item 2a of Lemma 5.0.12 to continue Eq. (5.10) as:

  $$\mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}) \geq \sum_{l_4 \leq l < l_5} \mathbb{1}\left(\mathsf{Err}_{\mathsf{Last}^C_{i-1}(l)} > 0 \wedge \mathsf{Pre}_{i-1} \in \mathsf{S}_{\mathsf{Last}^C_{i-1}(l)}\right).$$

  To finish, we show that each term above is 1 and use Eq. (5.11). Fix $l_4 \leq l < l_5$ and define $i' = \mathsf{Last}_{i-1}^C(l)$. We first claim that $\mathsf{Pre}_{i'} < l_3$. Indeed, if not, we have by Item 1 of Lemma 5.0.15 that $\mathsf{Z}^A_{\mathsf{Last}^C_{i'}(l_3)}.\mathtt{hash} = \mathsf{Z}^B_{\mathsf{Last}^C_{i'}(l_3)}.\mathtt{hash}$. Using Item 2a of Lemma 5.0.12, this gives $\mathsf{Z}^A_{\mathsf{Last}^C_{i-1}(l_3)}.\mathtt{hash} = \mathsf{Z}^B_{\mathsf{Last}^C_{i-1}(l_3)}.\mathtt{hash}$, a contradiction.

  We now use $\mathsf{Pre}_{i'} < l_3$ to show that the term corresponding to l is 1. For this, we need to show that $\mathsf{Err}_{i'} > 0$ and $\mathsf{Pre}_{i-1} \in \mathsf{S}_{i'}$. For the former, note by Definition 5.0.1 that $i' \in \boxed{C9}$ and $\mathsf{Z}_{i'}^C.\mathsf{bn} = l$ and thus, if $\mathsf{Err}_{i'} = 0$, we can derive a contradiction from Items 2 and 5 of Lemma 5.0.20. For the latter, we use Eq. (5.8) and show that $\mathsf{Pre}_{i-1} = \arg\min_{\mathsf{M} \in \mathsf{CMP}_{i'}^C} \mathsf{ddl}(\mathsf{M}.\mathsf{bn})$.

  Note that $\mathsf{Pre}_{i-1} \in \mathsf{CMP}_{i'}^C$ follows from Definition 5.0.2 and Lemma 5.0.21 and it is enough to show that $\mathsf{ddl}(\mathsf{Pre}_{i-1}) \leq \mathsf{ddl}(l')$ for all $l' \in \mathsf{CMP}_{i'}^C$. Fix an arbitrary l' and note from Eq. (5.7) that $l' \leq \mathsf{Pre}_{i'} < l_3$. Use Lemma 5.0.8 to conclude that either $\mathsf{ddl}(l') \geq \mathsf{ddl}(\mathsf{Pre}_{i-1})$ or $\mathsf{ddl}(l') \leq l_4$. As Item 1b of Lemma 5.0.12 implies the latter cannot happen, we are done.

- **When $\mathsf{Z}^A_{\mathsf{Last}^C_{i-1}(l_3)}.\mathtt{hash} = \mathsf{Z}^B_{\mathsf{Last}^C_{i-1}(l_3)}.\mathtt{hash}$:** Let $i_3 = \mathsf{Last}_{i-1}^C(l_3)$. Applying Item 1a of Lemma 5.0.12, we get that $\mathsf{Z}^A_{\mathsf{Last}^A_{i_3}(\mathsf{Z}^A_{i_3}.\mathsf{bn})}.\mathtt{hash} = \mathsf{Z}^B_{\mathsf{Last}^B_{i_3}(\mathsf{Z}^B_{i_3}.\mathsf{bn})}.\mathtt{hash}$. Now, applying Lemma 5.0.14, we get $\mathsf{Z}^A_{i_3}.\mathsf{bn} = \mathsf{Z}^B_{i_3}.\mathsf{bn}$. As the definition of i_3 implies this common value must be l_3, we get from Definition 5.0.2 that $\mathsf{Pre}_{i_3} = l_3 = \mathsf{Z}^A_{i_3}.\mathsf{bn} = \mathsf{Z}^B_{i_3}.\mathsf{bn}$.

  Let $\overline{C} \in \{A, B\}$ be the unique element different from C. By Eq. (5.8) and Item 2a of Lemma 5.0.12, we can continue Eq. (5.10) as:

  $$\mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}) \geq \sum_{i_3 \leq i' < i} \mathbb{1}\left(\mathsf{Err}_{i'} > 0 \wedge \mathsf{Pre}_{i-1} = \max_{\mathsf{M} \in \mathsf{MP}_{i'}^{\overline{C}}} \mathsf{M}.\mathsf{bn}\right).$$

  Now, we claim that:

  CLAIM 5.0.1. *For all $i_3 \leq i' < i$, if $\mathsf{Z}_{i'}^{\overline{C}}.\mathsf{bn} = \mathsf{Pre}_{i'} = l_2$, then $\mathsf{Pre}_{i-1} = \max_{\mathsf{M} \in \mathsf{MP}_{i'}^{\overline{C}}} \mathsf{M}.\mathsf{bn}$.*

Proof. Due to Item 2 of Lemma 5.0.15 we have $\mathsf{Pre}_{i-1} \in \mathcal{P}_{i'}$ implying due to $\mathsf{Z}^{\overline{C}}_{i'}.\mathsf{bn} = l_2$ and Definition 5.0.2 that $\mathsf{Pre}_{i-1} \leq \max_{\mathsf{M} \in \mathsf{MP}^{\overline{C}}_{i'}} \mathsf{M}.\mathsf{bn}$. Thus, it suffices to show that $\max_{\mathsf{M} \in \mathsf{MP}^{\overline{C}}_{i'}} \mathsf{M}.\mathsf{bn} \leq \mathsf{Pre}_{i-1}$. We show this by contradiction. If this is not true, then, by Observation 5.0.3, there exists $\mathsf{M} \in \mathsf{MP}^{\overline{C}}_{i'}$ such that $\mathsf{Pre}_{i-1} = l_1 < \mathsf{M}.\mathsf{bn} < l_2 = \mathsf{Pre}_{i'}$. By Item 1 of Lemma 5.0.15, this means that $\mathsf{Last}^A_{i'}(\mathsf{M}.\mathsf{bn}) = \mathsf{Last}^B_{i'}(\mathsf{M}.\mathsf{bn})$ and with Item 2a of Lemma 5.0.12, we get that they are both equal to $\mathsf{Last}^C_{i-1}(\mathsf{M}.\mathsf{bn})$. However, by Item 1b of Lemma 5.0.12, this means that $l_3 = \mathsf{Z}^{\overline{C}}_{i_3}.\mathsf{bn} < \mathsf{ddl}(\mathsf{M}.\mathsf{bn})$, a contradiction to Lemma 5.0.7. ∎

Using Claim 5.0.1, we get:

$$\mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}) \geq \sum_{i_3 \leq i' < i} \mathbb{1}\Big(\mathsf{Err}_{i'} > 0 \wedge \mathsf{Z}^{\overline{C}}_{i'}.\mathsf{bn} = \mathsf{Pre}_{i'} = l_2\Big).$$

Now, note that for all $i_3 \leq i' < i$ such that $\mathsf{Z}^{\overline{C}}_{i'}.\mathsf{bn} = \mathsf{Pre}_{i'} = l_2$, $i' \in \boxed{\overline{C}14} \setminus \boxed{\overline{C}16}$ only if $\mathsf{Err}_{i'} > 0$. Indeed, $i' \in \boxed{\overline{C}14} \setminus \boxed{\overline{C}16}$ implies $l_2 = \mathsf{Z}^{\overline{C}}_{i'}.\mathsf{bn} = \mathsf{Z}^{\overline{C}}_{i'-1}.\mathsf{bn} = b^{\overline{C}}_{i'} \geq b'^{\overline{C}}_{i'}$. If $\mathsf{Err}_{i'} = 0$, we can use Items 1 and 2 of Lemma 5.0.20 to continue as $l_2 \geq b^C_{i'} = \mathsf{Z}^C_{i'-1}.\mathsf{bn} \geq \mathsf{Z}^C_{i'}.\mathsf{bn}$. As the last term is at least l_3 by Item 2a of Lemma 5.0.12, this is a contradiction. We get:

$$\mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}) \geq \sum_{i_3 \leq i' < i} \mathbb{1}\Big(i' \in \boxed{\overline{C}14} \setminus \boxed{\overline{C}16} \wedge \mathsf{Z}^{\overline{C}}_{i'}.\mathsf{bn} = \mathsf{Pre}_{i'} = l_2\Big).$$

We now claim that:

Claim 5.0.2. *There exists $i_3 < i^* < i$ such that $i^* \in \boxed{\overline{C}16}$ and $\mathsf{Z}^{\overline{C}}_{i^*-1}.\mathsf{bn} = \mathsf{Pre}_{i^*-1} = l_2$.*

We defer the proof of Claim 5.0.2 to later but show here why it finishes our proof of Lemma 5.0.23. Let i^* be as promised by Claim 5.0.2 and use Claim 5.0.1 to conclude that $\mathsf{Pre}_{i-1} = \max_{\mathsf{M} \in \mathsf{MP}^{\overline{C}}_{i^*-1}} \mathsf{M}.\mathsf{bn}$. As $i^* \in \boxed{\overline{C}16}$, this means that $\mathsf{Z}^{\overline{C}}_{i^*}.\mathsf{bn} = \mathsf{Pre}_{i-1}$. By Lemma 5.0.13, we have $E^{\overline{C}}_{i^*-1} = l_2 - l_1 - 1$.

Note now from Algorithm 1 that whenever Z or MP change in any iteration, the value of E is reset to 0 (Line 9 and Lemma 5.0.13). Moreover, the value of E increases by at most 1 in any iteration and is increased only when a party executes Line 14 but not Line 16. This means the only way we can have $E^{\overline{C}}_{i^*-1} = l_2 - l_1 - 1$ is if there are $l_2 - l_1 - 1$ iterations $a_1 < a_2 < \cdots < a_{l_2-l_1-1} < i^*$, all of them in the set $\boxed{\overline{C}14} \setminus \boxed{\overline{C}16}$ such that $\Big(\mathsf{Z}^{\overline{C}}_{i'}, \mathsf{MP}^{\overline{C}}_{i'}\Big) = \Big(\mathsf{Z}^{\overline{C}}_{i^*-1}, \mathsf{MP}^{\overline{C}}_{i^*-1}\Big)$ for all $a_1 \leq i' < i^*$. By the definition of i_3, this can only happen if $i_3 < a_1$. Thus, we are done if we can show that $\mathsf{Pre}_{i'} = l_2$ for all $a_1 \leq i' < i^*$. Fix such an i'. As $\mathsf{Z}^{\overline{C}}_{i'}.\mathsf{bn} = \mathsf{Z}^{\overline{C}}_{i^*-1}.\mathsf{bn} = l_2$ and we have Observation 5.0.5, it is enough to show that that $\mathsf{Pre}_{i'} \geq \mathsf{Pre}_{i^*-1} = l_2$. This is because of Item 2 of Lemma 5.0.15.

We finish by showing Claim 5.0.2.

Proof. [Proof of Claim 5.0.2] Define $i_3 < i^* < i$ to be the smallest such that $\mathsf{Pre}_{i^*} < l_2$ and $\mathsf{Pre}_{i^*-1} \geq l_2$. Recall that $\mathsf{Pre}_{i_3} = l_3$ and $\mathsf{Pre}_{i-1} = l_1$ and therefore i^* is well defined. Now, as $\mathsf{Pre}_{i^*-1} \leq \mathsf{Z}^C_{i^*-1}.\mathsf{bn} < l_5$ by Item 2 of Lemma 5.0.15 and Corollary 5.0.2 says that $\lfloor \mathsf{Pre}_{i^*-1} \rfloor_{\mathsf{ev}(\mathsf{Pre}_{i^*-1})} \leq \mathsf{Pre}_{i^*}$, we have that $\mathsf{Pre}_{i^*} < l_2$ is possible only if $\mathsf{Pre}_{i^*-1} = l_2$.

We now claim that it is enough to show that $\mathsf{Z}^{\overline{C}}_{i^*}.\mathsf{bn} < l_2$. Indeed, if $\mathsf{Z}^{\overline{C}}_{i^*}.\mathsf{bn} < l_2$, then $\mathsf{Z}^{\overline{C}}_{i^*}.\mathsf{bn} < \mathsf{Pre}_{i^*-1} \leq \mathsf{Z}^{\overline{C}}_{i^*-1}.\mathsf{bn}$ implying that $i^* \in \boxed{\overline{C}16}$ and $l_2 > \mathsf{Z}^{\overline{C}}_{i^*}.\mathsf{bn} = \max_{\mathsf{M} \in \mathsf{MP}^{\overline{C}}_{i^*-1}} \mathsf{M}.\mathsf{bn}$. From the latter and the fact that $\mathsf{Pre}_{i^*-1} = l_2$, we also get using Definition 5.0.2 that $\mathsf{Z}^{\overline{C}}_{i^*-1}.\mathsf{bn} = l_2$, as desired.

Finally, we show $\mathsf{Z}^{\overline{C}}_{i^*}.\mathsf{bn} < l_2$. Let $\mathsf{M}^A \in \mathsf{MP}^{\star A}_{i^*-1}$, $\mathsf{M}^B \in \mathsf{MP}^{\star B}_{i^*-1}$ be as in Definition 5.0.2. As $\mathsf{Pre}_{i^*} < \mathsf{Pre}_{i^*-1} = l_2$, there exists $C^* \in \{A, B\}$ such that $\mathsf{M}^{C^*} \in \mathsf{MP}^{\star C^*}_{i^*-1} \setminus \mathsf{MP}^{\star C^*}_{i^*}$. By Lemma 5.0.10 we either have $\mathsf{Z}^{C^*}_{i^*}.\mathsf{bn} \geq \mathsf{ddl}(l_2)$ or $\mathsf{Z}^{C^*}_{i^*}.\mathsf{bn} < l_2$. The former is impossible as $\mathsf{ddl}(l_2) > \mathsf{ddl}(l_1)$ (due to Observation 5.0.2 and Lemma 5.0.6) and we have Item 2 of Lemma 5.0.15. Additionally, by Item 2a of Lemma 5.0.12, the latter is possible only if $C^* = \overline{C}$ and we are done. ∎

□

LEMMA 5.0.24. *Let $C \in \{A, B\}$ and $i \in \boxed{C16}$ be such that $\mathsf{Z}^C_{i-1}.\mathsf{bn} = \mathsf{Pre}_{i-1}$. It holds that:*

$$\mathsf{Pre}_{i-1} - \mathsf{Pre}_i \leq 2 \cdot \left(\mathsf{Z}^C_{i-1}.\mathsf{bn} - \mathsf{Z}^C_i.\mathsf{bn}\right) + 8 \cdot \mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}).$$

Proof. We start by defining:

$$\lambda_1 = \mathsf{Pre}_{i-1} - \mathsf{Pre}_i \qquad \lambda_2 = \mathsf{Z}^C_{i-1}.\mathsf{bn} - \mathsf{Z}^C_i.\mathsf{bn} \qquad l_1 = \mathsf{Pre}_{i-1} - \frac{1}{2} \cdot \lambda_1.$$

As the lemma is trivial otherwise, we assume that $2\lambda_2 < \lambda_1$. By Corollary 5.0.2, we continue as $2\lambda_2 < \lambda_1 \leq \frac{1}{2} \cdot \mathsf{ev}(\mathsf{Pre}_{i-1})$. As $i \in \boxed{C16}$ and $\mathsf{Z}^C_{i-1}.\mathsf{bn} = \mathsf{Pre}_{i-1} > 0$, we have $\mathsf{MP}^{\star\,C}_i \subseteq \mathsf{MP}^C_{i-1}$ by Algorithm 1. By Line 20, it follows that there exists $z_1, z_2 \geq 0$ such that for all $j \in [2]$ we have $\mathsf{Pre}_{i-1} - \lambda_j = \lfloor \mathsf{Pre}_{i-1} \rfloor_{2^{z_j}} - 2^{z_j}$. As $2\lambda_2 < \lambda_1 \leq \frac{1}{2} \cdot \mathsf{ev}(\mathsf{Pre}_{i-1})$, this is only possible if for all $j \in [2]$, we have $\lambda_j = 2^{z_j}$ is a power of 2. It follows that:

$$\mathsf{ddl}\left(\mathsf{Z}^C_i.\mathsf{bn}\right) = \mathsf{Pre}_{i-1} + \lambda_2 \quad \mathsf{ddl}(l_1) = \mathsf{Pre}_{i-1} + \frac{1}{2} \cdot \lambda_1 \quad l_1 = \left\lfloor \mathsf{Z}^C_i.\mathsf{bn} \right\rfloor_{\frac{1}{2} \cdot \lambda_1} = \lfloor \mathsf{Pre}_{i-1} - 1 \rfloor_{\frac{1}{2} \cdot \lambda_1}.$$

Now use $0 < 2\lambda_2 < \lambda_1$ and Lemma 5.0.17 to get that there exists $\overline{C} \in \{A, B\}$ such that $\mathsf{M}.\mathsf{bn} \neq l_1$ for all $\mathsf{M} \in \mathsf{MP}^{\star\overline{C}}_i$. By Item 1c of Lemma 5.0.12, we get that $\overline{C} \neq C$. Using, Item 1 of Lemma 5.0.15, define $i_1 = \mathsf{Last}^C_{i-1}(\mathsf{Pre}_{i-1}) = \mathsf{Last}^{\overline{C}}_{i-1}(\mathsf{Pre}_{i-1})$. From Corollary 5.0.4 and Eq. (5.8), we get:

$$\mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}) \geq \sum_{i_1 < i' < i} \mathbb{1}(\mathsf{Err}_{i'} > 0 \wedge \mathsf{Pre}_{i-1} \in \mathsf{S}_{i'}) \geq \sum_{i_1 < i' < i} \mathbb{1}(\mathsf{Err}_{i'} > 0 \wedge \mathsf{Pre}_{i-1} = \mathsf{Pre}_{i'}).$$

By Lemma 5.0.20, note that for all $i_1 < i' < i$, we have that $i' \in \boxed{\overline{C}9}$ and $\mathsf{Z}^{\overline{C}}_{i'}.\mathsf{bn} > \mathsf{Z}^C_{i'}.\mathsf{bn}$ implies that $\mathsf{Err}_{i'} > 0$. We get:

$$\mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}) \geq \sum_{i_1 < i' < i} \mathbb{1}\left(i' \in \boxed{\overline{C}9} \wedge \mathsf{Z}^{\overline{C}}_{i'}.\mathsf{bn} > \mathsf{Z}^C_{i'}.\mathsf{bn} \wedge \mathsf{Pre}_{i-1} = \mathsf{Pre}_{i'}\right).$$

Next, use Item 1b of Lemma 5.0.12 to get $\mathsf{Z}^C_{i'}.\mathsf{bn} < \mathsf{ddl}\left(\mathsf{Z}^C_i.\mathsf{bn}\right) = \mathsf{Pre}_{i-1} + \lambda_2$ for all $i_1 \leq i' < i$. We get:

$$\mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}) \geq \sum_{i_1 < i' < i} \mathbb{1}\left(i' \in \boxed{\overline{C}9} \wedge \mathsf{Z}^{\overline{C}}_{i'}.\mathsf{bn} \geq \mathsf{Pre}_{i-1} + \lambda_2 \wedge \mathsf{Pre}_{i-1} = \mathsf{Pre}_{i'}\right).$$

We now claim that for all $i_1 < i' < i$, if $\mathsf{Z}^{\overline{C}}_{i'}.\mathsf{bn} \geq \mathsf{Pre}_{i-1} + \frac{3}{2} \cdot \lambda_2$, then $\mathsf{Pre}_{i-1} = \mathsf{Pre}_{i'}$. Fix such an i'. As Item 2 of Lemma 5.0.15 says that $\mathsf{Pre}_{i-1} \leq \mathsf{Pre}_{i'}$ and we know $\mathsf{Pre}_{i'} \leq \mathsf{Z}^C_{i'}.\mathsf{bn} < \mathsf{Pre}_{i-1} + \lambda_2$ from the argument above, all we have to show is that it is impossible for $\mathsf{Pre}_{i-1} < \mathsf{Pre}_{i'} < \mathsf{Pre}_{i-1} + \lambda_2$ to hold. Indeed, if this holds we use the fact that λ_2 is a power of 2 and $\lambda_2 < \mathsf{ev}(\mathsf{Pre}_{i-1})$ together with Lemma 5.0.7 to get $\mathsf{ddl}(\mathsf{Pre}_{i'}) \leq \mathsf{Pre}_{i-1} + \frac{3}{2} \cdot \lambda_2 \leq \mathsf{Z}^{\overline{C}}_{i'}.\mathsf{bn}$. This contradicts Item 2 of Lemma 5.0.15. We get:

$$\mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}) \geq \sum_{i_1 < i' < i} \mathbb{1}\left(i' \in \boxed{\overline{C}9} \wedge \mathsf{Z}^{\overline{C}}_{i'}.\mathsf{bn} \geq \mathsf{Pre}_{i-1} + \frac{3}{2} \cdot \lambda_2\right).$$

Now, note by definition of i_1 that $\mathsf{Z}^{\overline{C}}_{i_1} = \mathsf{Pre}_{i-1}$. Moreover $\mathsf{Z}^{\overline{C}}.\mathsf{bn}$ increases by at most 1 in any iteration and only increases in iterations in $\boxed{\overline{C}9}$. We get:

$$\mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}) \geq \max_{i_1 < i' < i} \mathsf{Z}^{\overline{C}}_{i'}.\mathsf{bn} - \mathsf{Pre}_{i-1} - \frac{3}{2} \cdot \lambda_2 + 1.$$

We now claim that $\max_{i_1 < i' < i} \mathsf{Z}^{\overline{C}}_{i'}.\mathsf{bn} \geq \mathsf{ddl}(l_1) - 1 = \mathsf{Pre}_{i-1} + \frac{1}{2} \cdot \lambda_1 - 1$. Assuming this claim for now and recalling that $2\lambda_2 < \lambda_1 \implies 4\lambda_2 \leq \lambda_1$ as λ_1, λ_2 are powers of 2, we get:

$$\mathsf{Pre}_{i-1} - \mathsf{Pre}_i = \lambda_1 \leq 4\lambda_1 - 10\lambda_2 = 2\lambda_2 + 8 \cdot \left(\frac{1}{2} \cdot \lambda_1 - \frac{3}{2} \cdot \lambda_2\right) \leq 2\lambda_2 + 8 \cdot \mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}).$$

It remains to show the claim. For this, first note from Observation 5.0.4 that is suffices to show that $\max_{i_1<i'\le i} \mathsf{Z}_{i'}^{\overline{C}}.\mathsf{bn} \ge \mathsf{ddl}(l_1)$. If there exists $\mathsf{M} \in \mathsf{MP}_{i-1}^{\overline{C}} \setminus \mathsf{MP}^{\star}{}_{i}^{\overline{C}}$ such that $\mathsf{M}.\mathsf{bn} = l_1$, then we are done by Lemma 5.0.10, so we assume otherwise. As $\mathsf{Z}_{i-1}^{\overline{C}}.\mathsf{bn} \ge \mathsf{Pre}_{i-1} > l_1$, this means that there does not exist $\mathsf{M} \in \mathsf{MP}^{\star}{}_{i-1}^{\overline{C}} \setminus \mathsf{MP}^{\star}{}_{i}^{\overline{C}}$ such that $\mathsf{M}.\mathsf{bn} = l_1$. Using our choice of $\overline{C}$, we further get that there does not exist $\mathsf{M} \in \mathsf{MP}^{\star}{}_{i-1}^{\overline{C}}$ such that $\mathsf{M}.\mathsf{bn} = l_1$.

Now, using Item 2c of Lemma 5.0.12, we get that $\max_{\mathsf{Last}_{i-1}^{\overline{C}}(l_1)<i'<i} \mathsf{Z}_{i'}^{\overline{C}}.\mathsf{bn} \ge \mathsf{ddl}(l_1)$. It therefore suffices to show that $\max_{\mathsf{Last}_{i-1}^{\overline{C}}(l_1)<i'\le i_1} \mathsf{Z}_{i'}^{\overline{C}}.\mathsf{bn} < \mathsf{ddl}(l_1)$. By definition of i_1, we have $\mathsf{Z}_{i_1}^{\overline{C}}.\mathsf{bn} < \mathsf{ddl}(l_1)$ and therefore it suffices to show that $\max_{\mathsf{Last}_{i-1}^{\overline{C}}(l_1)<i'<i_1} \mathsf{Z}_{i'}^{\overline{C}}.\mathsf{bn} < \mathsf{ddl}(l_1)$. Use Item 2a of Lemma 5.0.12 to get that this is exactly the same as showing $\max_{\mathsf{Last}_{i_1-1}^{\overline{C}}(l_1)<i'<i_1} \mathsf{Z}_{i'}^{\overline{C}}.\mathsf{bn} < \mathsf{ddl}(l_1)$. However, this follows from $l_1 = \lfloor \mathsf{Pre}_{i-1} - 1 \rfloor_{\frac{1}{2}\cdot\lambda_1}$, the definition of i_1 and Items 1b and 1c of Lemma 5.0.12. ∎

5.2.10 A Potential Function We are now ready to define our potential function Φ. For $i \in \{0\} \cup [M']$, we define:

$$\Phi_i = 1000 \cdot \mathsf{Err}_{\le i} + 11 \cdot \mathsf{Pre}_i - 100 \cdot \sum_{l \in \mathsf{CMP}_i} \mathsf{Tax}_i(l) + \sum_{C \in \{A,B\}} 5 \cdot \left(\mathsf{K}_i^C \cdot E_i^C - \mathsf{Z}_i^C.\mathsf{bn}\right). \tag{5.12}$$

In the lemmas that follow, we shall show that Φ increases by at least a constant in every iteration.

LEMMA 5.0.25. *Let $i \in [M']$ be such that $\mathsf{Pre}_i < \mathsf{Pre}_{i-1}$. It holds that:*

$$\Phi_{i-1} + 1000 \cdot \mathsf{Err}_i - 900 \le \Phi_i.$$

Proof. As $\mathsf{Pre}_i < \mathsf{Pre}_{i-1}$, we have $\mathsf{Pre}_{i-1} \in \mathsf{CMP}_{i-1} \setminus \mathsf{CMP}_i$ (using Lemma 5.0.21) which by Lemma 5.0.22 gives:

$$\sum_{l \in \mathsf{CMP}_i} \mathsf{Tax}_i(l) \le 5 + \sum_{l \in \mathsf{CMP}_{i-1}} \mathsf{Tax}_{i-1}(l) - \mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}). \tag{5.13}$$

We also have, by Lemma 5.0.19, for all $C \in \{A, B\}$:

$$\mathsf{K}_{i-1}^C \cdot E_{i-1}^C - \mathsf{Z}_{i-1}^C.\mathsf{bn} \le 6 + \mathsf{K}_i^C \cdot E_i^C - \mathsf{Z}_i^C.\mathsf{bn}. \tag{5.14}$$

Next, let $\mathsf{M}^A \in \mathsf{MP}^{\star}{}_{i-1}^{A}$, $\mathsf{M}^B \in \mathsf{MP}^{\star}{}_{i-1}^{B}$ be as in Definition 5.0.2. As $\mathsf{Pre}_i < \mathsf{Pre}_{i-1}$, there exists $C \in \{A, B\}$ such that $\mathsf{M}^C \in \mathsf{MP}^{\star}{}_{i-1}^{C} \setminus \mathsf{MP}^{\star}{}_{i}^{C}$. We assume $C = A$ without loss of generality and consider the following cases:

- **When $\mathsf{M}^A \in \mathsf{MP}_{i-1}^{A} \setminus \mathsf{MP}^{\star}{}_{i}^{A}$:** In this case, use Lemma 5.0.10 to get that $i \in \boxed{A9}$ and $\mathsf{Z}_i^A.\mathsf{bn} = \mathsf{ddl}\big(\mathsf{M}^A.\mathsf{bn}\big) = \mathsf{ddl}(\mathsf{Pre}_{i-1})$. We have from Lemma 5.0.23 that:

  $$\mathsf{Pre}_{i-1} - \mathsf{Pre}_i \le 8 \cdot \mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}) + 8. \tag{5.15}$$

  Multiplying Eq. (5.13) by 100, Eq. (5.15) by 11, Eq. (5.14) by 5 (for all $C \in \{A, B\}$) and adding, we get

  $$\begin{aligned} 100 \cdot \sum_{l \in \mathsf{CMP}_i} \mathsf{Tax}_i(l) + 11 \cdot (\mathsf{Pre}_{i-1} - \mathsf{Pre}_i) + \sum_{C \in \{A,B\}} 5 \cdot \left(\mathsf{K}_{i-1}^C \cdot E_{i-1}^C - \mathsf{Z}_{i-1}^C.\mathsf{bn}\right) \\ \le 900 + 100 \cdot \sum_{l \in \mathsf{CMP}_{i-1}} \mathsf{Tax}_{i-1}(l) + \sum_{C \in \{A,B\}} 5 \cdot \left(\mathsf{K}_i^C \cdot E_i^C - \mathsf{Z}_i^C.\mathsf{bn}\right). \end{aligned}$$

  Rearranging and using Eq. (5.12) finishes the proof.

- **When $\mathsf{M}^A = \mathsf{Z}_{i-1}^A \notin \mathsf{MP}^{\star}{}_{i}^{A}$:** In this case, we have $\mathsf{Z}_{i-1}^A.\mathsf{bn} = \mathsf{Pre}_{i-1}$, and using Lemma 5.0.10, also have that $\mathsf{Z}_i^A.\mathsf{bn} < \mathsf{Z}_{i-1}^A.\mathsf{bn}$ implying $i \in \boxed{A16}$. We have from Lemma 5.0.24 that:

  $$\mathsf{Pre}_{i-1} - \mathsf{Pre}_i \le 2 \cdot \left(\mathsf{Z}_{i-1}^A.\mathsf{bn} - \mathsf{Z}_i^A.\mathsf{bn}\right) + 8 \cdot \mathsf{Tax}_{i-1}(\mathsf{Pre}_{i-1}). \tag{5.16}$$

Again using $i \in \boxed{A16}$ and $\mathsf{Z}_i^A.\mathsf{bn} < \mathsf{Z}_{i-1}^A.\mathsf{bn}$, we have by Lemma 5.0.13 that $E_i^A = 0$ and:

$$E_{i-1}^A + 1 = \mathsf{Z}_{i-1}^A.\mathsf{bn} - \max_{\mathsf{M} \in \mathsf{MP}_{i-1}^A} \mathsf{M}.\mathsf{bn} = \mathsf{Z}_{i-1}^A.\mathsf{bn} - \mathsf{Z}_i^A.\mathsf{bn}.$$

As Eq. (5.6) implies that $\mathsf{K}_{i-1}^A = -5$, we get from $E_i^A = 0$ that:

$$\begin{aligned}\mathsf{K}_{i-1}^A \cdot E_{i-1}^A - \mathsf{Z}_{i-1}^A.\mathsf{bn} &\le -5 \cdot \left(\mathsf{Z}_{i-1}^A.\mathsf{bn} - \mathsf{Z}_i^A.\mathsf{bn} - 1\right) - \mathsf{Z}_{i-1}^A.\mathsf{bn} \\ &= 5 - 6 \cdot \left(\mathsf{Z}_{i-1}^A.\mathsf{bn} - \mathsf{Z}_i^A.\mathsf{bn}\right) - \mathsf{Z}_i^A.\mathsf{bn} \\ &= 5 - 6 \cdot \left(\mathsf{Z}_{i-1}^A.\mathsf{bn} - \mathsf{Z}_i^A.\mathsf{bn}\right) + \mathsf{K}_i^A \cdot E_i^A - \mathsf{Z}_i^A.\mathsf{bn}.\end{aligned}$$

Multiplying this by 5, Eq. (5.16) by 11, Eq. (5.14) for $C = B$ by 5, Eq. (5.13) by 100 and adding, we get:

$$\begin{aligned}11 \cdot \mathsf{Pre}_{i-1} + 100 \cdot \sum_{l \in \mathsf{CMP}_i} \mathsf{Tax}_i(l) + \sum_{C \in \{A,B\}} 5 \cdot \left(\mathsf{K}_{i-1}^C \cdot E_{i-1}^C - \mathsf{Z}_{i-1}^C.\mathsf{bn}\right) \\ \le 555 + 11 \cdot \mathsf{Pre}_i + 100 \cdot \sum_{l \in \mathsf{CMP}_{i-1}} \mathsf{Tax}_{i-1}(l) + \sum_{C \in \{A,B\}} 5 \cdot \left(\mathsf{K}_i^C \cdot E_i^C - \mathsf{Z}_i^C.\mathsf{bn}\right)\end{aligned}$$

Rearranging and using Eq. (5.12) finishes the proof.

□

LEMMA 5.0.26. *Let $i \in [M']$. We have:*

$$\Phi_{i-1} + 1 \le \Phi_i.$$

Proof. We divide the proof into the following cases:

- **When $\mathsf{Err}_i = 0$:** In this case, we have:

  (Eq. (5.12) and $\mathsf{Err}_i = 0$)
  $$\Phi_{i-1} = 1000 \cdot \mathsf{Err}_{\le i} + 11 \cdot \mathsf{Pre}_{i-1} - 100 \cdot \sum_{l \in \mathsf{CMP}_{i-1}} \mathsf{Tax}_{i-1}(l) + \sum_{C \in \{A,B\}} 5 \cdot \left(\mathsf{K}_{i-1}^C \cdot E_{i-1}^C - \mathsf{Z}_{i-1}^C.\mathsf{bn}\right)$$

  (Lemma 5.0.22 and $\mathsf{Err}_i = 0$)
  $$\le 1000 \cdot \mathsf{Err}_{\le i} + 11 \cdot \mathsf{Pre}_{i-1} - 100 \cdot \sum_{l \in \mathsf{CMP}_i} \mathsf{Tax}_i(l) + \sum_{C \in \{A,B\}} 5 \cdot \left(\mathsf{K}_{i-1}^C \cdot E_{i-1}^C - \mathsf{Z}_{i-1}^C.\mathsf{bn}\right)$$

  (Corollary 5.0.3)
  $$\le -1 + 1000 \cdot \mathsf{Err}_{\le i} + 11 \cdot \mathsf{Pre}_i - 100 \cdot \sum_{l \in \mathsf{CMP}_i} \mathsf{Tax}_i(l) + \sum_{C \in \{A,B\}} 5 \cdot \left(\mathsf{K}_i^C \cdot E_i^C - \mathsf{Z}_i^C.\mathsf{bn}\right)$$

  (Eq. (5.12)) $\le \Phi_i - 1.$

- **When $\mathsf{Err}_i = 1$ and $\mathsf{Pre}_i < \mathsf{Pre}_{i-1}$:** In this case, the lemma follows by Lemma 5.0.25.

- **When $\mathsf{Err}_i = 1$ and $\mathsf{Pre}_{i-1} \le \mathsf{Pre}_i$:** In this case, we have:

  (Eq. (5.12) and $\mathsf{Pre}_{i-1} \le \mathsf{Pre}_i$)
  $$\Phi_{i-1} \le 1000 \cdot \mathsf{Err}_{<i} + 11 \cdot \mathsf{Pre}_i - 100 \cdot \sum_{l \in \mathsf{CMP}_{i-1}} \mathsf{Tax}_{i-1}(l) + \sum_{C \in \{A,B\}} 5 \cdot \left(\mathsf{K}_{i-1}^C \cdot E_{i-1}^C - \mathsf{Z}_{i-1}^C.\mathsf{bn}\right)$$

  (Lemma 5.0.22)
  $$\le 500 + 1000 \cdot \mathsf{Err}_{<i} + 11 \cdot \mathsf{Pre}_i - 100 \cdot \sum_{l \in \mathsf{CMP}_i} \mathsf{Tax}_i(l) + \sum_{C \in \{A,B\}} 5 \cdot \left(\mathsf{K}_{i-1}^C \cdot E_{i-1}^C - \mathsf{Z}_{i-1}^C.\mathsf{bn}\right)$$

  (Lemma 5.0.19)
  $$\le 600 + 1000 \cdot \mathsf{Err}_{<i} + 11 \cdot \mathsf{Pre}_i - 100 \cdot \sum_{l \in \mathsf{CMP}_i} \mathsf{Tax}_i(l) + \sum_{C \in \{A,B\}} 5 \cdot \left(\mathsf{K}_i^C \cdot E_i^C - \mathsf{Z}_i^C.\mathsf{bn}\right)$$

  (As $\mathsf{Err}_i = 1$) $\le -100 + \Phi_i.$

□

COROLLARY 5.0.5. *We have* $M \le \mathsf{Pre}_{M'}$.

Proof. Note from Eq. (5.12) and Lemma 5.0.26 that $\Phi_0 = 0$ and $\Phi_{M'} \ge M'$. This means that

$$\begin{aligned} M' &\le 1000 \cdot \mathsf{Err}_{\le M'} + 11 \cdot \mathsf{Pre}_{M'} - 100 \cdot \sum_{l \in \mathsf{CMP}_{M'}} \mathsf{Tax}_{M'}(l) + \sum_{C \in \{A,B\}} 5 \cdot \left(\mathsf{K}^C_{M'} \cdot E^C_{M'} - \mathsf{Z}^C_{M'}.\mathsf{bn}\right) \\ &\le 1000 \cdot \mathsf{Err}_{\le M'} + 11 \cdot \mathsf{Pre}_{M'} + \sum_{C \in \{A,B\}} 5 \cdot \left(\mathsf{K}^C_{M'} \cdot E^C_{M'} - \mathsf{Z}^C_{M'}.\mathsf{bn}\right) \\ \text{(Lemma 5.0.18)} \qquad &\le 1000 \cdot \mathsf{Err}_{\le M'} + \mathsf{Pre}_{M'}. \end{aligned}$$

Next, we claim that $\mathsf{Err}_{\le M'} \le 10M \cdot \frac{\epsilon}{\epsilon'}$. Indeed, if not, we have by definition of Err that the number of corruptions made by the adversary is at least $10\epsilon T$ by Eq. (4.1), a contradiction. This means that $M' \le 10^4 M \cdot \frac{\epsilon}{\epsilon'} + \mathsf{Pre}_{M'}$ implying by Eq. (4.1) that $M \le \mathsf{Pre}_{M'}$. □

5.3 Finishing the Proof We now prove our main result.

Proof. [Proof of Theorem 3.2] Recall that due to Lemmas 5.0.1 and 5.0.2 and Observation 5.0.1 and the way we fixed the randomness in Eq. (5.5), all we have to show is that $\Pi'_{\mathsf{Adv}}(x^A, x^B) = \Pi(x^A, x^B)$. Let $\pi \in \{0,1\}^{2T}$ be the transcript containing the symbols sent by both the parties in a noiseless execution of Π when the inputs are x^A and x^B. Observe from Line 22 and the way we padded the protocol Π that $\Pi'_{\mathsf{Adv}}(x^A, x^B) = \Pi(x^A, x^B)$ follows if we show that there exists transcripts π^A, π^B that agree with π in the first $2T$ coordinates such that for all $C \in \{A, B\}$, party C goes to state $\mathsf{Z}^C_{M'}.\mathtt{state}$ when their input is x^C and they receive symbols as in π^C. To this end, define, for $C \in \{A, B\}$,

$$\pi^C = \sigma^C_{\mathsf{Last}^C_{M'}(1)} \| \sigma^C_{\mathsf{Last}^C_{M'}(2)} \| \dots \| \sigma^C_{\mathsf{Last}^C_{M'}\left(\mathsf{Z}^C_{M'}.\mathsf{bn}\right)}.$$

We first fix an arbitrary $C \in \{A, B\}$ and show that party C goes to state $\mathsf{Z}^C_{M'}.\mathtt{state}$ when their input is x^C and they receive symbols as in π^C. Due to Item 1a of Lemma 5.0.12, this follows from the following claim:

CLAIM 5.0.3. *For all* $0 \le l \le \mathsf{Z}^C_{M'}.\mathsf{bn}$*, party* C *goes to state* $\mathsf{Z}^C_{\mathsf{Last}^C_{M'}(l)}.\mathtt{state}$ *when their input is* x^C *and they receive the first* $2B^*l$ *symbols of* π^C.

Proof. Proof by induction on l. The base case $l = 0$ is straightforward. We show the result for $l > 0$ assuming it holds for $l - 1$. For this, define $i' = \mathsf{Last}^C_{M'}(l)$ and consider the iteration i'. From Item 2b of Lemma 5.0.12, we have $\mathsf{Z}^C_{i'-1} = \mathsf{Z}^C_{\mathsf{Last}^C_{M'}(l-1)}$. Using the induction hypothesis, we get that party C goes to state $\mathsf{Z}^C_{i'-1}.\mathtt{state}$ when their input is x^C and they receive the first $2B^*(l-1)$ symbols of π^C. To finish the proof, simply observe from Line 10, that if party C is in state $\mathsf{Z}^C_{i'-1}.\mathtt{state}$ with input x^C and receives $\sigma^C_{i'}$ (which are the next $2B^*$ symbols of π^C), then it goes to state $\mathsf{Z}^C_{i'}.\mathtt{state}$. □

It remains to show that the transcripts π^A, π^B agree with π in the first $2T$ coordinates. For this, note first that due to Lemma 5.0.14 and Item 1 of Lemma 5.0.15, we have for all $0 \le l \le \mathsf{Pre}_{M'}$ that $\mathsf{Last}^A_{M'}(l) = \mathsf{Last}^B_{M'}(l)$ and, using i_l to denote the common value, also have $l > 0 \implies \sigma^A_{i_l} = \sigma^B_{i_l}$. Due to Corollary 5.0.5, this in particular holds for all $l \in [M]$. Fix $l \in [M]$. As $\sigma^A_{i_l} = \sigma^B_{i_l}$, we have that this common value is the transcript generated when the parties execute B^* rounds of Π, starting from the states $\mathsf{Z}^A_{i_l-1}.\mathtt{state}$, $\mathsf{Z}^B_{i_l-1}.\mathtt{state}$ with inputs x^A, x^B respectively and leads the parties to update their states to $\mathsf{Z}^A_{i_l}.\mathtt{state}$ and $\mathsf{Z}^B_{i_l}.\mathtt{state}$ (respectively). From Item 2b of Lemma 5.0.12, we have $\mathsf{Z}^C_{i_l-1} = \mathsf{Z}^C_{i_{l-1}}$ for all $C \in \{A, B\}$ and thus, we have that $\sigma^A_{i_l} = \sigma^B_{i_l}$ is the transcript generated when the parties execute B^* rounds of Π, starting from the states $\mathsf{Z}^A_{i_{l-1}}.\mathtt{state}$, $\mathsf{Z}^B_{i_{l-1}}.\mathtt{state}$ with inputs x^A, x^B respectively and leads the parties to update their states to $\mathsf{Z}^A_{i_l}.\mathtt{state}$ and $\mathsf{Z}^B_{i_l}.\mathtt{state}$. As $\mathsf{Z}^A_{i_0}.\mathtt{state} = \mu^A_0$ and $\mathsf{Z}^B_{i_0}.\mathtt{state} = \mu^B_0$ are the starting states of Alice and Bob respectively, we get that π^A, π^B agree with π in the first $2B^*M = 2T$ coordinates. □

References

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

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

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

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

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

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

[7] T.-H. Hubert Chan, Zhibin Liang, Antigoni Polychroniadou, and Elaine Shi. Small memory robust simulation of client-server interactive protocols over oblivious noisy channels. In *Symposium on Discrete Algorithms (SODA)*, pages 2349–2365, 2020.

[8] Gil Cohen and Shahar Samocha. Palette-alternating tree codes. In *35th Computational Complexity Conference (CCC 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

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

[10] Klim Efremenko, Bernhard Haeupler, Yael Tauman Kalai, Pritish Kamath, Gillat Kol, Nicolas Resch, and Raghuvansh R Saxena. Circuits resilient to short-circuit errors. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 582–594, 2022.

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

[12] Ran Gelles and Bernhard Haeupler. Capacity of interactive communication over erasure channels and channels with feedback. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1296–1311. SIAM, 2014.

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

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

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

[16] Venkatesan Guruswami and Piotr Indyk. Linear-time encodable/decodable codes with near-optimal rate. *IEEE Transactions on Information Theory*, 51(10):3393–3400, 2005.

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

[18] Bernhard Haeupler and Nicolas Resch. Coding for interactive communication with small memory and applications to robust circuits. *arXiv*, abs/1805.06872v1, 2018.

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

[20] Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM journal on computing*, 22(4):838–856, 1993.

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

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

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

[24] 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.

[25] Michael Sipser and Daniel A Spielman. Expander codes. *IEEE transactions on Information Theory*, 42(6):1710–1722, 1996.