Electronic Colloquium on Computational Complexity, Report No. 154 (2021)

Explicit Binary Tree Codes with Sub-Logarithmic Size Alphabet

Inbar Ben Yaacov[*] Gil Cohen[†] Tal Yankovitz [‡]

November 10, 2021

Abstract

Since they were first introduced by Schulman (STOC 1993), the construction of tree codes remained an elusive open problem. The state-of-the-art construction by Cohen, Haeupler and Schulman (STOC 2018) has constant distance and $(\log n)^e$ colors for some constant $e > 1$ that depends on the distance, where n is the depth of the tree. Insisting on a constant number of colors at the expense of having vanishing distance, Gelles, Haeupler, Kol, Ron-Zewi, and Wigderson (SODA 2016) constructed a distance $\Omega(\frac{1}{\log n})$ tree code.

In this work we improve upon these prior works and construct a distance-δ tree code with $(\log n)^{O(\sqrt{\delta})}$ colors. This is the first construction of a constant distance tree code with sub-logarithmic number of colors. Moreover, as a direct corollary we obtain a tree code with a constant number of colors and distance $\Omega\left(\frac{1}{(\log\log n)^2}\right)$, exponentially improving upon the above-mentioned work by Gelles *et al.*

[*]Department of Computer Science, Tel Aviv University, Israel. Funded by the Azrieli Faculty Fellowship. inbarb1@mail.tau.ac.il

[†]Department of Computer Science, Tel Aviv University, Israel. Supported by ERC starting grant 949499 and by the Israel Science Foundation (grant number 1569/18). gil@tauex.tau.ac.il

[‡]Department of Computer Science, Tel Aviv University, Israel. Supported by ERC starting grant 949499. talyankovitz@mail.tau.ac.il

ISSN 1433-8092

Contents

1 Introduction

Coding theory addresses the problem of communication over an imperfect channel. In the classic setting [Sha48, Ham50], Alice wishes to communicate a message to Bob over a channel that may introduce errors. The question then is: how should Alice encode her message so that if the amount of errors is not excessive, Bob can recover her message? To this end the notion of an error-correcting code was introduced. A function $C\colon \Sigma^k \to \Sigma^n$ is an *error-correcting code* with distance δ if for every distinct $x, y \in \Sigma^k$, the respective images $C(x), C(y)$ have relative Hamming distance at least δ. The rate of information transmission $\rho = \frac{k}{n}$ and the fraction of errors corrected (roughly $\frac{\delta}{2}$) are competing quantities. Among the most basic questions in coding theory is to obtain explicit *asymptotically good codes*, that is, codes over fixed Σ with constant distance $\delta > 0$ and constant rate $\rho > 0$. By "explicit" we mean that C can be evaluated in time $\mathrm{poly}(n)$. Justensen [Jus72] was the first to devise such an explicit construction. Since then, several explicit constructions have appeared, e.g., [TVZ82, SS96].

While error-correcting codes can be used to solve the problem of sending a single message from Alice to Bob over an imperfect channel, in some settings, the two parties interact with each other, sending multiple messages where a message depends on previous messages that were exchanged. Interactive coding addresses the subtler problem of enabling such dynamic interaction over an imperfect channel. In this far more challenging setting, standard codes do not offer a satisfactory solution. We refer the interested reader to the excellent survey by Gelles [Gel17] but do not discuss interactive coding schemes further.

Tree codes are powerful combinatorial structures, defined by Schulman [Sch93, Sch96] as key ingredients for achieving interactive coding schemes. They play a role analogous to the one error-correcting codes take in the single message setting. Tree codes, as their name suggests, are trees with a certain distance property. To give the formal definition, we set some notation. Let T be the infinite complete rooted binary tree that is endowed with an edge coloring from some ambient color set, or alphabet, Σ. For vertices u, v of equal depth let w be their least common ancestor and denote the distance, in edges, from u to w by ℓ. Let $p_u, p_v \in \Sigma^\ell$ be the sequences of colors on the path from w to u and to v, respectively. We define $h(u, v)$ to be the relative Hamming distance between p_u and p_v. Informally, $h(u, v)$ measures the distance between the two color sequences obtained by following the paths from the root to each of u and v, excluding the "non-interesting" common prefix. A tree code is any coloring that has a lower bound on this quantity.

Formally,

Definition 1.1 (Tree codes [Sch93])**.** *Let T be the complete infinite rooted binary tree. The tree T, together with an edge-coloring of T by a color set Σ is called a* binary tree code with distance δ *if for every pair of vertices u, v with equal depth, it holds that $h(u, v) \geq \delta$.*

A second definition of tree codes in the literature refers not to one infinite tree but rather to a family $(T_n)_{n\in\mathbb{N}}$ where T_n is a depth n rooted complete binary tree. The family is called a tree code with distance δ if every T_n has distance δ as defined above. It is clear that an infinite tree code with distance δ induces such a family with the same distance, simply by truncating the infinite tree at the different depths. Unless stated otherwise, when referring to a tree code we mean a single infinite tree.

A priori, it is not at all clear that there exists a tree code with distance $\delta > 0$. Three different proofs were provided by Schulman, showing that for any constant $\delta < 1$ there exists a tree code with alphabet size $|\Sigma| = O_\delta(1)$ achieving distance δ. More recently, based on Schulman's ideas, it was shown that there is a tree code with only four colors and positive distance (in particular, distance $\delta > 0.136$) [CS20] and, moreover, three colors do not suffice to guarantee distance $\delta > 0$. All of these proofs rely on the probabilistic method and thus are non-explicit. The problem of constructing tree codes has drawn substantial attention [Sch94, Bra12, MS14, Pud16, GHK$^+$16, CHS18, NW20, BH20, BYCN20], but has endured as a difficult challenge.

Given this difficulty, it is natural to construct, for a given constant distance parameter $\delta > 0$, a tree code with number of colors $c = c(n)$ that may depend on the depth n. The goal is to obtain an asymptotically slowly-growing function c. Note that constructing a tree code with $c(n) = 2^n$ colors is trivial. Indeed, having so many colors at our disposal, we can encode the entire path leading to a vertex on the edge preceding it, yielding distance $\delta = 1$. In an unpublished manuscript, Evans, Klugerman and Schulman [Sch94] constructed a tree code with $c(n) = n^{O_\delta(1)}$ colors. The first improvment was obtained a couple of years ago by Cohen, Haeupler and Schulman [CHS18] who achieved $c(n) = (\log n)^e$ where $e > 1$ is some function of $'delta$. See [NW20] for alternative constructions achieving the same parameters as well as decoding algorithms, and [BH20] for an account relating [CHS18] and [Pud16]. Two explicit constructions with a constant number of colors $c = O_\delta(1)$ were suggested [MS14, BYCN20] though their correctness relies on unproven, yet seemingly plausible, conjectures.

The analog question of constructing a tree code family with a constant number of colors and a slowly deteriorating distance was first considered by Gelles, Haeupler, Kol, Ron-Zewi, and Wigderson [GHK+16]. The authors gave an explicit family of tree codes $(T_n)_{n\in\mathbb{N}}$ where the number of colors in every tree in the family is bounded by a universal constant c and the distance $\delta(n)$ of T_n satisfies $\delta(n) = \Omega(\frac{1}{\log n})$.

1.1 Our results

The CHS construction, as well as the variants suggested by [NW20], require $\omega(\log n)$ colors. More precisely, the number of colors takes the form $(\log n)^{1+f(\delta)}$ where f is a strictly positive function that vanishes as $\delta \to 0$. Hence, prior to this work, no construction with sub-logarithmic number of colors was known, for any constant distance $\delta > 0$. The main result of this work is a construction that beats this logarithmic threshold.

Theorem 1.2 (Main result)**.** *There exists a universal constant $\delta_0 > 0$ such that the following holds. For every $\delta \le \delta_0$ there exists an explicit binary tree code with distance δ and $(\log n)^{O(\sqrt{\delta})}$ colors at depth n.*

The following is an immediate corollary.

Corollary 1.3. *There exists an explicit family of binary tree codes $(T_n)_{n\in\mathbb{N}}$ with a constant number of colors such that T_n has distance $\Omega\left(\frac{1}{(\log\log n)^2}\right)$.*

Corollary 1.3 improves exponentially upon the construction by Gelles *et al.* [GHK+16] who, recall, obtained an explicit tree code family with a constant number of colors and distance guarantee $\Omega(\frac{1}{\log n})$.

Before we present the ideas that go into the proof of Theorem 1.2 (see Section 2) we describe another minor but useful contribution of this work. As mentioned, tree codes are defined in two different ways: either as a single infinite tree or as an infinite family of finite trees. As mentioned, it is well-known that the first definition implies the latter. More precisely, an infinite tree code induces a tree code family. To the best of our knowledge, prior to this work the converse was not known to hold, and indeed the distinction was made explicit in several prior works. As a side contribution, we give in Appendix A a very simple argument for the converse. This allows us to consider only finite tree codes, and as a consequence, makes our construction and analysis somewhat simpler.

2 Proof overview

In this section we give an informal overview of our tree code construction that is given by Theorem 1.2. To this end we first give an alternative, equivalent, definition of a tree code as was used by [CHS18] (see Section 2.1), and in Section 2.2 we recall some facts about the CHS construction. Starting from Section 2.3 we present our construction and its analysis.

2.1 An alternative definition of a tree code

Tree codes, as their name suggests, are trees with a certain distance property. However, in this paper, we use an equivalent definition of tree codes that we find more convenient to work with. We start by giving some basic definitions. First, a function $f \colon \Sigma_{\mathsf{in}}^n \to \Sigma_{\mathsf{out}}^n$ is said to be *online* if for every $i \in [n]$ and $x \in \Sigma_{\mathsf{in}}^n$, $f(x)_i$ is determined by $x_1, \ldots, x_i$. Second, for $x, y \in \Sigma^n$ we denote by $\mathsf{dist}(x, y)$ the Hamming distance between x and y. Third, for a pair of distinct $x, y \in \Sigma^n$, we define $\mathsf{split}(x, y)$ as the least integer $s \in [n]$ such that $x_s \neq y_s$. We turn to give the alternative definition of a tree code. Given our reduction from infinite to finite tree codes, we focus on the latter.

Definition 2.1. *An online function* $\mathsf{TC} \colon \Sigma_{\mathsf{in}}^n \to \Sigma_{\mathsf{out}}^n$ *is a* tree code with distance δ *if for every distinct* $x, y \in \Sigma_{\mathsf{in}}^n$*, with* $s = \mathsf{split}(x, y)$*, and every* $\ell \in \{0, 1, \ldots, n - s\}$*,*

$$\mathsf{dist}\left(\mathsf{TC}(x)_{[s, s+\ell]}, \mathsf{TC}(y)_{[s, s+\ell]}\right) \geq \delta(\ell + 1).$$

We refer to n *as the* depth *of* TC*, and to* $\Sigma_{\mathsf{in}}, \Sigma_{\mathsf{out}}$ *as the input alphabet and output alphabet, respectively.*

The equivalence between Definition 1.1 and Definition 2.1 can be easily verified. We point out that the terms "depth" and "split" are coming from the original point of view on tree codes, namely, Definition 1.1. Indeed, the depth is simply the depth of the tree and the split is the level at which the pair of paths diverge. We borrow this terminology even though we do not explicitly view tree codes as trees from this point on. Note further that $|\Sigma_{\mathsf{out}}|$ corresponds to the number of colors used by the tree. A tree code $\mathsf{TC} \colon \Sigma_{\mathsf{in}}^n \to \Sigma_{\mathsf{out}}^n$ is called binary if $\Sigma_{\mathsf{in}} = \{0, 1\}$. We say that TC is *explicit* if it can be evaluated on every input $x \in \Sigma_{\mathsf{in}}^n$ in polynomial time in the bit complexity of x.

2.2 The parameters of the CHS construction

To construct their binary tree code, in [CHS18] the authors first construct a large arity tree code, namely taking Σ_{in} large. Then, in the second step, they reduce the alphabet to binary. More precisely, in the first step, a depth ℓ tree code with arity 2^m is constructed with roughly $2^{2m+\ell}$ colors and distance $\frac{1}{2}$. This by itself is a very wasteful binary tree code. Indeed, setting $m = 1$ the tree code requires more than 2^ℓ colors at depth ℓ which, as mentioned, can be obtained trivially by recording the entire path leading to a node.

Nonetheless, the advantage of this construction is that m, ℓ are, in some sense, decoupled. Indeed, for comparison, the ideas in [Sch94] can be used to yield a construction with $\ell^{O(m)}$ colors. Although the latter has a much better dependence on ℓ, only for constant ℓ is the number of colors polynomial in 2^m. Taking advantage of this property, to construct a depth n binary tree code, [CHS18] split the n bits to $\ell = \sqrt{n}$ consecutive blocks of length $m = \sqrt{n}$ each and apply the 2^m-arity tree code, identifying each block with a symbol from a 2^m-size alphabet. Of course, each output symbol can only be recorded at the location of the next block. Sweeping some issues under the rug, for every bit read, one writes $\frac{2m+\ell}{m} = 3$ bits. However, a "lag" of length $m = \sqrt{n}$ is incurred which is resolved by a recursive construction applied to each block. As there are $\log_2 \log n$ levels of recursion, and since we write 3 bits per level, the number of colors required is $(\log n)^3$. Again, this discussion ignores some issues that further increase the number of colors. For example, when "transforming" the 2^m-size alphabet symbol into bits one needs to apply an error correcting code so that a nonzero symbol from the 2^m-size alphabet will contribute many nonzero bits.

2.3 Proof strategy - improving the dependence on the depth

The number of bits written per m bits read in the CHS construction is $2m + \ell$. The first summand, $2m$, is "natural" in that it encodes the fact that one writes two bits for every bit read. The second summand is "unnatural" and it is an artifact of the [CHS18] construction. Indeed, computational aspects aside, one does not have a dependence on the tree code's depth.

Our strategy is thus very natural - improve the dependence on ℓ, up to the point of eliminating it entirely. Towards this end, we show how to significantly improve the dependence on ℓ while slightly deteriorating the dependence on m and, importantly, without any distance deterioration. First, we break up the different components that effect the number of colors. Specifically, we say that a depth-ℓ tree code over $\{0,1\}^m$ is an (α, β, γ) tree code if it writes at most $(1+\alpha)m + \beta\ell^\gamma$ bits per (m-bit) symbol read. That is, the tree code takes the form

$$\mathsf{TC} : (\{0,1\}^m)^\ell \to \left(\{0,1\}^{(1+\alpha)m+\beta\ell^\gamma}\right)^\ell .$$

Our key technical contribution is devising an efficient way of transforming an (α, β, γ) tree code to an $(\alpha + O(\sqrt{\delta}), \beta + O(\sqrt{\delta}), \frac{\gamma}{2+\gamma})$ tree code. The emphasis is on the improved γ parameter, and in particular on the fact that the "new" γ is at most half the original one. A slight drawback of our transformation is that it does not work over every alphabet size (and, in particular, it does not work for binary tree codes), however, it is guaranteed to work in the regime $m = \Omega(\log \ell)$. This turns out to be a sufficiently small alphabet so that binary tree codes can be obtained with little cost and effort.

The initial tree code to which we apply our transformation is CHS which, in our notations, is a $(1, 1, 1)$ tree code. In fact, before doing so, we observe that one can tweak the CHS construction, so when aiming to have distance δ it is in fact a $(\delta, \delta, 1)$ tree code. Applying our transformation once to the latter already gives an $(O(\sqrt{\delta}), O(\sqrt{\delta}), \frac{1}{3})$ tree code with distance δ. Applying the transformation to the resulted tree code gives an $(O(\sqrt{\delta}), O(\sqrt{\delta}), \frac{1}{7})$ tree code, again, with distance δ, where the hidden constants get larger. Continuing in this manner, applying the transformation iteratively, for r times, each time to the previously computed tree code yields an $(O(r\sqrt{\delta}), O(r\sqrt{\delta}), \frac{1}{2^{r+1}-1})$ tree code with distance δ. For eliminating the dependence on ℓ altogether we set $r \approx \log \log \ell$ and obtain a distance-δ tree code of the form

$$\mathsf{TC} : (\{0,1\}^m)^\ell \to \left(\{0,1\}^{O(\sqrt{\delta} \cdot \log \log \ell) m}\right)^\ell .$$

Finally, for reducing the alphabet to binary we use a simple and well-known transformation that has almost no cost in parameters as long as $m = O(\log \ell)$. Luckily, as mentioned, our transformation works in the regime $m \geq c \cdot \log n$ for some universal constant c.

2.4 The transformation

We now sketch how to transform an (α, β, γ) tree code to an $(\alpha + O(\sqrt{\delta}), \beta + O(\sqrt{\delta}), \frac{\gamma}{2+\gamma})$ tree code. Concretely, we would like to obtain a depth-ℓ tree code over the alphabet $\{0,1\}^m$. Denote by $x = (x_1, \ldots, x_\ell) \in (\{0,1\}^m)^\ell$ the message to be encoded. Write $\ell = \ell_1 \ell_2$ for some parameters ℓ_1, ℓ_2 to be chosen later on. We think of the message x written in matrix form with row length ℓ_1 (and thus ℓ_2 rows) row by row, from left to right. That is, the first row consists of $x^{(1)} = (x_1, \ldots, x_{\ell_1})$, recorded from left to right; the second row consists of the next ℓ_1 symbols $x^{(2)} = (x_{\ell_1+1}, \ldots, x_{2\ell_1})$, and so forth until the last row $x^{(\ell_2)}$.

The transformation starts by applying the (α, β, γ) tree code that is given as input to the transformation,

$$\mathsf{TC}_{\mathsf{row}} : (\{0,1\}^m)^{\ell_1} \to (\{0,1\}^{m_1})^{\ell_1}$$

which is assumed to have distance δ, to each row r so as to obtain $y^{(r)} = \mathsf{TC}_{\mathsf{row}}\big(x^{(r)}\big) \in (\{0,1\}^{m_1})^{\ell_1}$, where $m_1 = (1+\alpha)m + \beta\ell_1^{\gamma}$. This by itself guarantees that from the split until the end of its row, the relative distance is δ. However, as $\mathsf{TC}_{\mathsf{row}}$ is applied to each row without taking into account the information from prior rows, there is no distance guarantee in future rows.

Tensoring? By inspecting the literature on other types of codes, in particular locally testable and locally decodable codes, one can see that tensoring is a natural way to overcome this issue of the distance effecting only a single row. Viewing $y^{(1)}, \ldots, y^{(\ell_2)}$ as the ℓ_2 rows of an $\ell_2 \times \ell_1$ matrix over $\{0,1\}^{m_1}$, let $w^{(1)}, \ldots, w^{(\ell_1)}$ be this matrix's columns. By tensoring we mean applying a second tree code $\mathsf{TC}_{\mathsf{col}} : (\{0,1\}^{m_1})^{\ell_2} \to (\{0,1\}^{m_2})^{\ell_2}$ to each column $w^{(c)}$. Intuitively, this should take care of the distance in future rows. However, as the tree codes we consider perform better when there is a large symbol-size-to-depth ratio, it turns out that despite some advantages entailed by tensoring, it is significantly better to consider each row as one large symbol. Moreover, unlike with tensoring, we can work with the original message and not with the (more expensive) output of the code $\mathsf{TC}_{\mathsf{row}}$ when applied to each row of the message.

Thus, at the second step we view each row of the message $x^{(r)}$, $r = 1, \ldots, \ell_2$, as a symbol in $\{0,1\}^{m\ell_1}$. Let

$$\mathsf{TC}_{\mathsf{col}} : (\{0,1\}^{m\ell_1})^{\ell_2} \to (\{0,1\}^{m_2\ell_1})^{\ell_2}$$

be an $(\alpha_{\mathsf{col}}, \beta_{\mathsf{col}}, \gamma_{\mathsf{col}})$ tree code with distance δ_{col}, and let $z_r = \mathsf{TC}_{\mathsf{col}}(x)_r$ for $r = 1, \ldots, \ell_2$. Of course, we wish to reduce these large symbols back to length ℓ_1 strings, and we want the transformation to have the property that a large nonzero symbol will correspond to a string with many nonzeros. To this end, let $\mathsf{C} : (\{0,1\}^{m_2})^{\ell_1} \to (\{0,1\}^{m_3})^{\ell_1}$ be a block code with distance δ_{C}. Note that we use a somewhat nonstandard code type in which the block length is equal to the message length and it is the output alphabet size that is larger than the input alphabet size. However, such a code can be easily obtained from standard codes.

Of course, we cannot output $\mathsf{C}(z_r) = \mathsf{C}(\mathsf{TC}_{\mathsf{col}}(x)_r)$ at row r as $\mathsf{TC}_{\mathsf{col}}$ is online only in the "resolution" of full rows. Further, C itself is not online. Instead, we output $\mathsf{C}(z_r)$ in row $r+1$. To summarize, the construction so far is defined by

$$\mathsf{TC}(x)_{r,c} = \left(\mathsf{TC}_{\mathsf{row}}\left(x^{(r)}\right)_c, \mathsf{C}\left(\mathsf{TC}_{\mathsf{col}}(x)_{r-1}\right)_c\right). \tag{2.1}$$

2.4.1 Analyzing the redundancy so far

Although TC as defined in Equation (2.1) is not yet a tree code as we have no sufficient bound on its distance (as we explain, and resolve, in Section 2.4.2), it is instructive to analyze its redundancy already at this point. First, we need to choose $\mathsf{TC}_{\mathsf{col}}$. There are two natural candidates: CHS or the tree code that was obtained in the previous iteration. However, it turns out that if we wish to maintain the distance then, between these two options, we are forced to take CHS. Thus, $\mathsf{TC}_{\mathsf{col}}$ is a $(\delta_{\mathsf{col}}, \delta_{\mathsf{col}}, 1)$ tree code with distance δ_{col}. As we apply $\mathsf{TC}_{\mathsf{row}}$ to obtain the first component in the output of TC, we write

$$m_1 = (1+\alpha)m + \beta\ell_1^{\gamma}$$

bits in the first component. As for the second component, by the choice of $\mathsf{TC}_{\mathsf{col}}$, we get

$$m_2\ell_1 = (1+\alpha_{\mathsf{col}})m\ell_1 + \beta_{\mathsf{col}}\ell_2^{\gamma_{\mathsf{col}}} = (1+\delta_{\mathsf{col}})m\ell_1 + \delta_{\mathsf{col}}\ell_2,$$

and so $m_2 = (1+\delta_{\mathsf{col}})m + \delta_{\mathsf{col}}\frac{\ell_2}{\ell_1}$. The application of C then gives

$$m_3 = (1+\delta_{\mathsf{C}})m_2 \lesssim (1+\delta_{\mathsf{col}}+\delta_{\mathsf{C}})m + (1+\delta_{\mathsf{C}})\delta_{\mathsf{col}}\frac{\ell_2}{\ell_1},$$

where we used the fact that the code C, when set with distance δ_{C}, adds δ_{C} redundant symbols. Note that here we allow ourselves to ignore constant factors.

As in TC we output m_1+m_3 bits per m bits read, the concern that may come up is that the redundancy deteriorates by at least a factor of two (as, in particular, $m_1+m_3 \geq 2m$). However, by using a systematic code C and systematic tree codes $\mathsf{TC}_{\mathsf{row}}, \mathsf{TC}_{\mathsf{col}}$, we have that for every m bit read, TC outputs roughly

$$(1+\alpha+\delta_{\mathsf{col}}+\delta_{\mathsf{C}})m + (1+\delta_{\mathsf{C}})\delta_{\mathsf{col}}\frac{\ell_2}{\ell_1} + \beta\ell_1^{\gamma}$$

bits. Now, by taking ℓ_1, ℓ_2 such that $\ell_1^{\gamma} = \frac{\ell_2}{\ell_1}$ (recall also that $\ell = \ell_1\ell_2$) we deduce that TC outputs

$$(1+\alpha+\delta_{\mathsf{col}}+\delta_{\mathsf{C}})m + (\beta+2\delta_{\mathsf{col}})\ell^{\frac{\gamma}{2+\gamma}}$$

bits per m bits read. It is worth mentioning that the use of a systematic code so as to "pay" only for the redundant part entails some technical problems. Indeed, the redundant part of the code C is only recorded in the consecutive row, and so the redundancy is placed somewhat far from the systematic part.

As for the distance, although TC is not a tree code yet, we know that within a row the guaranteed distance is δ on the account of $\mathsf{TC}_{\mathsf{row}}$. As we do not want the transformation to deteriorate the distance, we must make sure that this is the bottleneck. When considering "sufficiently many" rows from the split, the fraction of nonzero output symbols of $\mathsf{TC}_{\mathsf{col}}$ is roughly δ_{col} and each contributes a δ_{C}-fraction of nonzero symbols after applying C. Thus, the distance cannot be better than $\min(\delta, \delta_{\mathsf{col}} \cdot \delta_{\mathsf{C}})$. By setting $\delta_{\mathsf{col}} = \delta_{\mathsf{C}} = \sqrt{\delta}$ the latter equals to δ, and we have that TC outputs at most

$$(1 + \alpha + 2\sqrt{\delta})m + (\beta + 2\sqrt{\delta})\ell^{\frac{\gamma}{2+\gamma}}$$

bits per m bits read. To summarize, the α, β components deteriorate by an additive $O(\sqrt{\delta})$ and, on the positive side, the dependence on γ improves by a factor of two (and, in fact, slightly better).

2.4.2 Fixing the transformation

To see the problem with the transformation as given so far by Equation (2.1), consider the case in which the split appears near the end of its row, say d entries before the end of the row. Then $\mathsf{TC}_{\mathsf{row}}$ will guarantee the existence of δd nonzeros within that row after the split. However, $\mathsf{TC}_{\mathsf{col}}$ and the code applied to it, C, are not guaranteed to output nonzeros at the beginning of the consecutive row, and it could be the case that all (or most) of the nonzero entries are recorded towards the end of that row. Hence, we have no meaningful guarantee on the distance.

A typical solution in such case is to use two tree codes in a "brick wall" like manner, namely, applying $\mathsf{TC}_{\mathsf{row}}$ twice - once on the rows as we have done so far, and a second time in which we apply $\mathsf{TC}_{\mathsf{row}}$ from the midpoint of a row to the midpoint of the next row. The first problem with this approach is that concatenating the two outputs will double the dependence on α. Here the use of a systematic tree code does not help as we cannot even afford the *redundancy* to double! A second problem with this simple approach is that the distance deteriorates by a factor of 2 and we cannot afford this lose. Indeed, recall that for constructing our tree code, we compose the transformation with itself for $\approx \log \log n$ times. Hence, a deterioration by a factor of 2 in the distance each time will result with a tree code having distance $\approx \frac{1}{\log n}$.

Thus, we are forced to come up with a different solution. The first idea that goes into our solution is to make sure that the smaller d (as defined above) is, the "faster" nonzero symbols are outputted at the next row, thus preventing the undesired phenomena described above of nonzero entries clustered towards the end of the row. To this end, we identify $\{0,1\}^m$ with the finite field of size 2^m, $\mathbb{F}_{2^m}$, in an arbitrary manner, and for row r, we define the polynomial $g_r(T) \in \mathbb{F}_{2^m}[T]$ by

$$g_r(T) = \sum_{i=1}^{\ell_1} x_i^{(r)} T^{\ell_1 - i}.$$

Then, at row $r+1$ we output, on top of the symbols that are given by Equation (2.1), the evaluations of g_r at arbitrary distinct elements of $\mathbb{F}_{2^m}$. Note that for this we need that $2^m \geq \ell_1$ which is the reason why our transformation only works over sufficiently large alphabet. As we are shooting for low deterioration on α, we do not output the evaluation at every entry but rather only every c entries, for some parameter c, and "spread" every evaluation over c consecutive blocks.

Let r be the row of the split. The point of this construction is that $\deg g_r = d$ and so, the smaller d is, the less zeros g_r has and, in particular, every prefix of the symbols outputted in row $r+1$ has at most d zeros. The worst case scenario is that the d zeros happen to be the first d field elements we evaluate at. These, recall, occupy the length cd prefix of row $r+1$ due to the above mentioned spread. In this case, the distance when considering a length $d+cd$ interval from the split is $\frac{\delta d}{(c+1)d} = \frac{\delta}{c+1}$. In fact, we also need to apply an error correcting code to the evaluations but we ignore this issue in this informal proof overview.

This seems to get us nowhere since, as already mentioned, we cannot afford to lose a constant factor in the distance. A key observation here is that for the above analysis we did not need the full-fledged tree code guarantee. Indeed, in the above calculation, we only considered the number of nonzeros *at the end* of row r which we bounded below by δd. Therefore, our second idea is to introduce a mechanism that guarantees that at the end of each row, the relative distance is Δ, for some parameter $\Delta > \delta$. This can be done using error correcting codes in a fairly simple manner and by paying $O(\Delta)$ in redundancy. Having done so, the above bound can be improved to $\frac{\Delta d}{(c+1)d}$. By choosing $c \approx \frac{1}{\sqrt{\delta}}$ and $\Delta \approx \sqrt{\delta}$ we guarantee no loss in distance as $\frac{\Delta d}{(c+1)d} \geq \delta$. In terms of the redundancy, the α component deteriorates by another additive $O\left(\frac{1}{c} + \Delta\right) = O(\sqrt{\delta})$ term.

To summarize, taking into account the contribution of $\mathsf{TC}_{\mathsf{row}}, \mathsf{TC}_{\mathsf{col}}$ and the code C applied to the latter, as well as the evaluation of the polynomials (g_r) and the mechanism that guarantees suffix distance Δ in each row, we have that at the expense of an additive $O(\sqrt{\delta})$ deterioration of the α, β parameters, the dependence on the γ parameter improves by (slightly more than) a factor of two, and further, the distance δ remains as is.

3 Preliminaries

All log are taken base 2. For positive integers ℓ, m, n such that $\ell \leq m$ and $\ell < n$, we define the intervals $[m] = \{1, \ldots, m\}$, $[\ell, m] = \{\ell, \ldots, m\}$, $[\ell, n) = \{\ell, \ldots, n-1\}$, and $[\ell, \infty) = \{\ell, \ell+1, \ell+2, \ldots\}$. Let i, j, k be integers such that $0 \leq i \leq j \leq n$ and $i < k \leq n$, and let Σ be some set. In contrast to the informal Section 2, from here on our indices will start from 0. Given a string $x = (x_0, \ldots, x_{n-1}) \in \Sigma^n$, we denote by $x_i \in \Sigma$ the i^{th} symbol of x and notate $x_{[i,j]} = (x_i, \ldots, x_j)$, $x_{[i,k)} = (x_i, \ldots, x_{k-1})$. For two strings x, y we denote their concatenation by $x \circ y$. We refer to the natural numbers as $\mathbb{N} = \{0, 1, 2, \ldots\}$.

Definition 3.1 (Hamming distance). *For two strings $x, x' \in \Sigma^n$, the* Hamming distance *between x and x' is defined to be $|\{\, i \in [0, n) \mid x_i \neq x'_i \,\}|$, and we denote it by $\mathsf{dist}(x, x')$. Further, the* relative distance *between x and x' is defined by $\frac{1}{n} \cdot \mathsf{dist}(x, x')$.*

Definition 3.2 (Split). *For two strings $x, x' \in \Sigma^n$, $x \neq x'$, we define the* split *of x and x' to be the minimal index i for which $x_i \neq x'_i$, and we denote it by $\mathsf{split}(x,x')$.*

Definition 3.3 (Systematic function). *Let k, n be positive integers and let $\Sigma_{\mathsf{in}}, \Sigma_{\mathsf{out}}$ be some sets. We say that a function $f : \Sigma_{\mathsf{in}}^k \to \Sigma_{\mathsf{out}}^n$ ($f : \Sigma_{\mathsf{in}}^{\mathbb{N}} \to \Sigma_{\mathsf{out}}^{\mathbb{N}}$) is* systematic *if there exist indices $i_1, \ldots, i_k \in [0, n)$ ($\{i_t\}_{t \in \mathbb{N}} \subseteq \mathbb{N}$) and indices $j_1, \ldots, j_k \in [0, |\Sigma_{\mathsf{out}}|)$ ($\{j_t\}_{t \in \mathbb{N}} \subseteq [0, |\Sigma_{\mathsf{out}}|)$) such that for every $x \in \Sigma_{\mathsf{in}}^k$ ($x \in \Sigma_{\mathsf{in}}^{\mathbb{N}}$), it holds that $(f(x)_{i_1})_{j_1}, \ldots, (f(x)_{i_k})_{j_k} = x$ ($((f(x)_{i_t})_{j_t})_{t \in \mathbb{N}} = x$).*

Definition 3.4 (Explicit function). *A function $f : \Sigma_{\mathsf{in}}^n \to \Sigma_{\mathsf{out}}^n$ ($f : \Sigma_{\mathsf{in}}^{\mathbb{N}} \to \Sigma_{\mathsf{out}}^{\mathbb{N}}$) is said to be* explicit *if for every $i \in [0, n)$ ($i \in \mathbb{N}$), $f(x)_i$ can be computed in time* $\mathrm{poly}(\log |\Sigma_{\mathsf{in}}|, n)$ $(\mathrm{poly}(\log |\Sigma_{\mathsf{in}}|, i+1))$.

3.1 Error-correcting block-codes

Definition 3.5 (Error-correcting codes). *A set $C \subseteq \Sigma^n$ is called a (ρ, δ)-*error-correcting code *if $\log_{|\Sigma|} |C| \geq \rho n$ and for any distinct $x, y \in C$, $\mathsf{dist}(x, y) \geq \delta n$. The maximal*

$\rho, \delta \in [0,1]$ *for which C is a (ρ,δ)-error-correcting code are called the* rate *and the* relative distance *of the code, respectively.*

Definition 3.6 (Reed-Solomon codes)**.** *Let $m, k, n \in \mathbb{N}$ such that $k \le n \le 2^m$. Let $\mathbb{F}_{2^m}$ be the field with 2^m elements and let $\alpha : \mathbb{F}_{2^m} \to \mathbb{F}_2^m$ be some $\mathbb{F}_2$-linear bijection. We define the (m,k,n)-*Reed-Solomon code *to be the following code $\mathsf{RS}_{(m,k,n)} \subseteq (\mathbb{F}_2^m)^n$,*

$$\mathsf{RS}_{(m,k,n)} = \{\, (\alpha(f(\gamma_1)), \dots, \alpha(f(\gamma_n))) \mid f \in \mathbb{F}_{2^m}[X], \deg(f) \le k-1 \,\},$$

*where $\gamma_1, \dots, \gamma_n \in \mathbb{F}_{2^m}$ are distinct. We define the (m,k,n)-*Reed-Solomon encoding*, $\mathsf{EncRS}_{(m,k,n)} : (\mathbb{F}_2^m)^k \to (\mathbb{F}_2^m)^n$ to be any fixed systematic encoding of $\mathsf{RS}_{(m,k,n)}$, i.e., a function which satisfies that $\mathsf{Img}(\mathsf{EncRS}_{(m,k,n)}) = \mathsf{RS}_{(m,k,n)}$ and that for every $x \in (\mathbb{F}_2^m)^k$, $\mathsf{EncRS}_{(m,k,n)}(x)_{[0,k)} = x$.*[1] *We define the (m,k,n)-*Reed-Solomon redundancy function *to be the following function $\mathsf{RedRS}_{(m,k,n)} : (\mathbb{F}_2^m)^k \to (\mathbb{F}_2^m)^{n-k}$, $\mathsf{RedRS}_{(m,k,n)}(x) = \mathsf{EncRS}_{(m,k,n)}(x)_{[k,n)}$.*

Fact 3.7. *For every $m, k, n \in \mathbb{N}$ such that $k \le n \le 2^m$, and $c, c' \in \mathsf{RS}_{(m,k,n)}$, $c \ne c'$, it holds that $\mathsf{dist}(c, c') \ge n - k + 1$.*

Throughout the paper we make use of length-preserving systematic codes. These can be obtained from standard codes in a fairly straightforward manner, however, for completeness, we give the proof of the following lemma in Appendix C.

Lemma 3.8. *For every positive integers m, k such that $k \le 2^{m-1}$ and every τ such that $\frac{1}{m} \le \tau \le 1$, there exists a systematic error-correcting code*

$$\mathsf{C} : (\{0,1\}^m)^k \to (\{0,1\}^m \times \{0,1\}^{\tau m})^k$$

with relative distance $\ge \frac{\tau}{4}$.

3.2 Tree codes

Definition 3.9 (Online function)**.** *A function $f : \Sigma_{\mathsf{in}}^n \to \Sigma_{\mathsf{out}}^n$ ($f : \Sigma_{\mathsf{in}}^{\mathbb{N}} \to \Sigma_{\mathsf{out}}^{\mathbb{N}}$) is said to be* online *if for every $x \in \Sigma_{\mathsf{in}}^n$ ($x \in \Sigma_{\mathsf{in}}^{\mathbb{N}}$) and $i \in [0,n)$ ($i \in \mathbb{N}$), $f(x)_i$ is determined by $x_0, \dots, x_i$.*

[1] Note that as Reed-Solomon is an MDS code, an encoding function that satisfies these properties exists.

Definition 3.10 (Tree code)**.** *An online function* $\mathsf{TC} : \Sigma_{\mathsf{in}}^n \to \Sigma_{\mathsf{out}}^n$ ($\mathsf{TC} : \Sigma_{\mathsf{in}}^{\mathbb{N}} \to \Sigma_{\mathsf{out}}^{\mathbb{N}}$) *is a depth-$n$ (infinite) tree code with distance δ if for every distinct $x, y \in \Sigma_{\mathsf{in}}^n$ ($x, y \in \Sigma_{\mathsf{in}}^{\mathbb{N}}$) such that $s = \mathsf{split}(x,y)$ and every $\ell \in [0, n-s)$, ($\ell \in \mathbb{N}$),*

$$\mathsf{dist}\Big(\mathsf{TC}(x)_{[s,s+\ell]}, \mathsf{TC}(y)_{[s,s+\ell]}\Big) \geq \delta(\ell+1).$$

A tree code is explicit *if* TC *is explicit according to Definition 3.4. A tree code is a* binary *tree code if* $\Sigma_{\mathsf{in}} = \{0,1\}$.

In our proof we will make use of the following standard fact.

Fact 3.11 (Reduction to binary alphabet)**.** *There exist universal constants* $c_{\mathsf{dist}} \in (0,1)$, $c_{\mathsf{rate}} > 1$ *for which the following holds. Let* $\mathsf{TC} : \Sigma^n \to (\Sigma^r)^n$ *be an explicit tree code with distance* $\delta \in (0,1)$*. Then* TC *can be transformed to a binary tree code*

$$\mathsf{TC}_{\mathsf{bin}} : \{0,1\}^{\log|\Sigma| n} \to \left(\{0,1\}^{c_{\mathsf{rate}} r}\right)^{\log|\Sigma| n}$$

with distance $c_{\mathsf{dist}}\delta$ *such that evaluating* $\mathsf{TC}_{\mathsf{bin}}$ *can be done in time* $\mathrm{poly}(n, |\Sigma|)$*. In particular, if* $|\Sigma| = \mathrm{poly}(n)$ *then* $\mathsf{TC}_{\mathsf{bin}}$ *is explicit.*

4 The γ-reducing transformation

In this section we present the transformation that improves upon the dependence on the γ component of a tree code as was informally discussed in Section 2. To this end, in Section 4.1 we introduce some preliminary definitions and assert some claims that we use in our transformation. Then, in Section 4.2 we present our transformation and its analysis.

4.1 Some preparations

For our transformation, it is essential to break down the number of colors a tree code uses to three components in terms of the tree's arity and depth. This enables us to divert the dependence of the number of colors between the three components.

Definition 4.1 (An (α, β, γ) tree code)**.** *Let $\alpha, \beta, \gamma \geq 0$ and let m, k, ℓ be positive integers. We say that a systematic tree code* $\mathsf{TC} : (\{0,1\}^m)^\ell \to \left(\{0,1\}^m \times \{0,1\}^k\right)^\ell$ *is an* (α, β, γ) tree code *if* $k \leq \alpha m + \beta \ell^\gamma$.

4.1.1 The CHS construction

In [CHS18], the authors provide a construction of a systematic tree code over the integers. For our transformation, we use a variation of their construction that has better dependence on α (and, less importantly, on β). An extended description of the CHS construction together with a proof for Claim 4.2 can be found in Appendix B.

Claim 4.2. *(A variation of the CHS construction with a better dependence on α) For every positive integers b, c, m, ℓ such that $b \leq c \leq 2^{\frac{m+\ell}{b}}$, there exists an explicit and systematic tree code*

$$\mathsf{TC}^{b,c}_{m,\ell} : (\{0,1\}^m)^\ell \to \left(\{0,1\}^m \times \{0,1\}^{\frac{m+\ell}{b}}\right)^\ell$$

with distance $\frac{c-b}{2c^2}$.

Corollary 4.3. *For every positive integers b, m, ℓ such that $b \leq 2^{\frac{m+\ell}{b}-1}$, there exists an explicit and systematic tree code*

$$\mathsf{TC}^{b,2b}_{m,\ell} : (\{0,1\}^m)^\ell \to \left(\{0,1\}^m \times \{0,1\}^{\frac{m+\ell}{b}}\right)^\ell$$

with distance $\frac{1}{8b}$.

Proof. Follows trivially from Claim 4.2 by setting $c = 2b$. □

4.1.2 More definitions and claims

For our transformation we make use of a tree code that, in addition to its regular notion of distance, has a stronger bound on its "suffix-distance". Here we define this property and argue for the existence of an efficient transformation that equips a tree code with any desired suffix-distance that is smaller than $\frac{1}{8}$, while preserving its (regular) distance and slightly increasing its alphabet size. We describe the transformation and prove its properties in Appendix D.

Definition 4.4 (Suffix distance)**.** *Let $f : \Sigma_{\mathsf{in}}^n \to \Sigma_{\mathsf{out}}^n$ be an online function and let $\Delta \in (0,1)$. We say that f has* suffix-distance Δ *if for every distinct $x, x' \in \Sigma_{\mathsf{in}}^n$ it holds that $\mathsf{dist}\big(f(x)_{[s,n)}, f(x')_{[s,n)}\big) \geq \Delta(n-s)$, where $s = \mathsf{split}(x,x')$.*

Definition 4.5 (Tree code with distance (δ, Δ))**.** *For $\delta, \Delta \in (0,1)$, we say that a tree code $\mathsf{TC} : \Sigma^n \to \Pi^n$ has* distance (δ, Δ) *if it has distance δ (as in the regular notion for tree codes) and suffix distance Δ.*

Claim 4.6 (Tree code transformation to obtain suffix-distance)**.** *Let* $\Delta \in \left(0, \frac{1}{8}\right]$, $\delta \in (0,1)$, $\alpha, \beta, \gamma \geq 0$, *and let* k, m, ℓ *be positive integers such that* $\ell \leq 2^m$. *Then, every systematic* (α, β, γ) *tree code*

$$\mathsf{TC} : (\{0,1\}^m)^\ell \to \left(\{0,1\}^m \times \{0,1\}^k\right)^\ell$$

with distance δ *can be efficiently transformed to a systematic* $(\alpha + 8\Delta, \beta, \gamma)$ *tree code*

$$\mathsf{TC}^\Delta : (\{0,1\}^m)^\ell \to \left(\{0,1\}^m \times \{0,1\}^{k'}\right)^\ell$$

with distance (δ, Δ). *We refer to* TC^Δ *as the* Δ-suffix-distance tree code of TC.

Lastly, we introduce some notions we require.

Definition 4.7 (Row-major matrix)**.** *Let* m, ℓ_1, ℓ_2 *be positive integers and set* $\ell = \ell_1 \ell_2$. *Let* $x \in (\{0,1\}^m)^\ell$ *be a string and let* $\mathcal{M}_x$ *be an* $\ell_2 \times \ell_1$ *matrix whose entries are elements in* $\{0,1\}^m$. *We say that* $\mathcal{M}_x$ *is the* $\ell_2 \times \ell_1$ row-major matrix of x *if the entries of* $\mathcal{M}_x$ *hold* x*'s symbols ordered by rows from left to right, top to bottom. That is, for every* $i \in [0, \ell_2)\,, j \in [0, \ell_1)\,,$

$$\mathcal{M}_x(i,j) = x_{i \cdot \ell_1 + j}.$$

Definition 4.8 (Spread)**.** *For integers* n, m, n' *such that* $n' \mid nm$ *we define a function* $\mathsf{spr}_{n'} : (\Sigma^m)^n \to \left(\Sigma^{\frac{nm}{n'}}\right)^{n'}$ *as follows. For every* $x \in (\Sigma^m)^n$, $\mathsf{spr}_{n'}(x)$ *is defined to be the string* $(y_0, \ldots, y_{n'-1}) \in \left(\Sigma^{\frac{mn}{n'}}\right)^{n'}$ *which satisfies that* $y_0 \circ \cdots \circ y_{n'-1} = x_0 \circ \cdots \circ x_{n-1}$ *as elements in* Σ^{mn}.

Claim 4.9. *For every* $x, x' \in (\Sigma^m)^n$ *and* $n \leq n'$, $n' \mid nm$,

$$\mathsf{dist}(\mathsf{spr}_{n'}(x)\,, \mathsf{spr}_{n'}(x')) \geq \frac{1}{2} \cdot \mathsf{dist}(x, x')\,.$$

Proof. Set $m' = \frac{nm}{n'}$ and note that $m' \leq m$. Further set $d = \mathsf{dist}(x, x')$ and let $j_1 < \cdots < j_d$ be indices in $[0, n)$ that satisfy $x_{j_i} \neq x'_{j_i}$. Let $t_1, \ldots, t_d \in [0, m)$ be indices for which $(x_{j_i})_{t_i} \neq (x'_{j_i})_{t_i}$. For every $i \in [d]$ set

$$\ell_i = \left\lfloor \frac{m j_i + t_i}{m'} \right\rfloor. \tag{4.1}$$

That is, ℓ_i is the index of the block of $\mathsf{spr}_{n'}(x)$ in which $(x_{j_i})_{t_i}$ resides. We have that $\mathsf{spr}_{n'}(x)_{\ell_i} \neq \mathsf{spr}_{n'}(x')_{\ell_i}$ for every $i \in [d]$. Note that it follows from Equation (4.1) using that $m' \leq m$, that if $i, i' \in [d]$ satisfy $i < i'$ and $\ell_i = \ell_{i'}$ then it must be that $j_{i'} = j_i + 1$, which in turn implies $i' = i+1$. It immediately follows that $|\{\ell_1, \ldots, \ell_d\}| \geq \frac{d}{2}$ and so $\mathsf{dist}(\mathsf{spr}_{n'}(x), \mathsf{spr}_{n'}(x')) \geq \frac{d}{2}$, as required. □

Claim 4.10. *Let k, n, n', m be positive integers such that $k < n \leq n'$ and $n' \mid (n-k)m$, and let Σ be some set. Then, for every $x, x' \in (\Sigma^m)^n$*

$$\mathsf{dist}\left(x_{[0,k)}, x'_{[0,k)}\right) + \mathsf{dist}\left(\mathsf{spr}_{n'}\left(x_{[k,n)}\right), \mathsf{spr}_{n'}\left(x'_{[k,n)}\right)\right) \geq \frac{1}{2}\mathsf{dist}(x, x').$$

Proof. Follows trivially from Claim 4.9. □

4.2 The γ-reducing transformation

Equipped with the definitions and claims from Section 4.1, we turn to describe our transformation. The main result proved in this section is the following.

Theorem 4.11. *There exist a constant $\delta_0 \in (0,1)$ and a transformation that given a systematic (α, β, γ) tree code*

$$\mathsf{TC}_{\mathsf{in}} : (\{0,1\}^m)^\ell \to \left(\{0,1\}^m \times \{0,1\}^k\right)^\ell \tag{4.2}$$

with distance δ, such that $m \geq \max\left\{2\log\ell, \frac{2}{\delta}\log\frac{1}{\delta}\right\}$, transforms it to a systematic $\left(\alpha + 267\sqrt{\min\{\delta, \delta_0\}}, \beta + 161\sqrt{\min\{\delta, \delta_0\}}, \frac{\gamma}{2+\gamma}\right)$ tree code

$$\mathsf{TC}_{\mathsf{out}} : (\{0,1\}^m)^{\ell^{2+\gamma}} \to \left(\{0,1\}^m \times \{0,1\}^{k_{\mathsf{out}}}\right)^{\ell^{2+\gamma}}$$

with distance $\delta_{\mathsf{out}} \geq \min\{\delta, \delta_0\}$. Moreover, if $\mathsf{TC}_{\mathsf{in}}$ is explicit then so is $\mathsf{TC}_{\mathsf{out}}$.

Proof. Let $\mathsf{TC}_{\mathsf{in}}$ be the tree code from Equation (4.2), let ℓ' be a positive integer to be determined later, and set $\ell_{\mathsf{out}} = \ell \cdot \ell'$. We use the following notations. For every $x \in (\{0,1\}^m)^{\ell_{\mathsf{out}}}$, set $\mathcal{M}_x$ to be the $\ell' \times \ell$ row-major matrix of x (see Definition 4.7). For every $(i,j) \in [0,\ell') \times [0,\ell)$, we denote the symbol in the $(i,j)^{\text{th}}$ entry of $\mathcal{M}_x$ by $\bar{x}_{i,j}$, and the length-ℓ substring that lies in the i^{th} row of $\mathcal{M}_x$ by $\bar{x}_i$.

Let $\mathbb{F}$ be the field of 2^m elements, and let $\varphi : \{0,1\}^m \to \mathbb{F}$, $\psi : \{0,\ldots,2^m-1\} \to \mathbb{F}$ be some bijections. For every $i \in [1,\ell')$ we define a polynomial $g_{x,i} \in \mathbb{F}[Z]$ by

$$g_{x,i}(Z) = \sum_{j=0}^{\ell-1} \varphi\left(\bar{x}_{i-1,j}\right) Z^{\ell-j-1}. \tag{4.3}$$

Namely, the coefficients of $g_{x,i}$ are the field elements corresponding to the symbols that lie in the $(i-1)^{\text{st}}$ row of $\mathcal{M}_x$, taken from right to left. For $i=0$ we set $g_{x,i}$ to be the zero polynomial.

4.2.1 Ingredients

Let $\Delta \in \left(0, \frac{1}{8}\right]$ and let b be a positive integer such that $b \le 2^{\frac{m}{b}-1}$. We set Δ, b later on (see Equation (4.4)). We use the following ingredients to construct $\mathsf{TC}_{\mathsf{out}}$.

- The Δ-suffix-distance tree code of $\mathsf{TC}_{\mathsf{in}}$,

$$\mathsf{TC}^{\Delta}_{\mathsf{in}} : (\{0,1\}^m)^\ell \to \left(\{0,1\}^m \times \{0,1\}^k \times \{0,1\}^{8\Delta m}\right)^\ell,$$

  which is obtained by Claim 4.6.

- A systematic $\left(\frac{1}{b}, \frac{1}{b}, 1\right)$ tree code

$$\mathsf{TC}_{\mathsf{col}} : \left(\{0,1\}^{m\ell}\right)^{\ell'} \to \left(\{0,1\}^{m\ell} \times \{0,1\}^{k'}\right)^{\ell'}$$

  with distance $\frac{1}{8b}$, given by Corollary 4.3. Observe that the existence of $\mathsf{TC}_{\mathsf{col}}$ is implied by the constraint $b \le 2^{\frac{m}{b}-1}$, that we impose on b.

- An explicit and systematic error-correcting code

$$\mathsf{C}_1 : \left(\{0,1\}^{\frac{m\ell+k'}{\ell}}\right)^\ell \to \left(\{0,1\}^{\frac{m\ell+k'}{\ell}} \times \{0,1\}^{\rho_1 \cdot \frac{m\ell+k'}{\ell}}\right)^\ell$$

  set with relative distance $\delta_1 = \frac{\rho_1}{4}$. The code C_1 exists per Lemma 3.8 per our assumption $m \ge 2\log\ell$. The parameter δ_1 (and therefore also ρ_1) will be set later on.

- The $\left(\frac{m}{b}, b, 2b\right)$-Reed-Solomon encoding

$$\mathsf{C}_2 : \left(\{0,1\}^{\frac{m}{b}}\right)^b \to \left(\{0,1\}^{\frac{m}{b}}\right)^{2b}$$

  with relative distances $\delta_2 = \frac{1}{2}$, given by Definition 3.6.

- The second component projections

$$\mathsf{R}_{\mathsf{col}} : \{0,1\}^{m\ell} \times \{0,1\}^{k'} \to \{0,1\}^{k'},$$

and

$$\mathsf{R}_{\mathsf{C}_1} : \{0,1\}^{\frac{m\ell+k'}{\ell}} \times \{0,1\}^{\rho_1 \cdot \frac{m\ell+k'}{\ell}} \to \{0,1\}^{\rho_1 \cdot \frac{m\ell+k'}{\ell}} .$$

- Finally, the set of polynomials $\{ g_{x,i} \mid i \in [0, \ell') \}$ as defined in Section 4.2.

4.2.2 The transformation

Let $x \in (\{0,1\}^m)^{\ell_{\mathsf{out}}}$ be a message to be encoded. Let $t \in [0, \ell_{\mathsf{out}})$ and set $(i,j) \in [0,\ell') \times [0,\ell)$ to be the indices corresponding to t, with respect to the row-major ordering as described in Definition 4.7. From now on, we identify t with its corresponding matrix indices. At time (i,j) the output of the resulting tree code $\mathsf{TC}_{\mathsf{out}} : (\{0,1\}^m)^{\ell_{\mathsf{out}}} \to \left(\{0,1\}^m \times \{0,1\}^{k_{\mathsf{out}}}\right)^{\ell_{\mathsf{out}}}$ consists of the following three parts.

- The first part of the output is taken to be

$$\mathsf{TC}^{\Delta}_{\mathsf{in}}(\bar{x}_i)_j \in \{0,1\}^m \times \{0,1\}^k \times \{0,1\}^{8\Delta m} ,$$

  namely, the j^{th} symbol of $\mathsf{TC}^{\Delta}_{\mathsf{in}}$, when evaluated on the i^{th} row of $\mathcal{M}_x$.

- For the second part, we compute

$$a_{i-1} = \mathsf{TC}_{\mathsf{col}}(x)_{i-1} \in \{0,1\}^{m\ell} \times \{0,1\}^{k'} ,$$

  where for the case of $i = 0$ we define $a_{-1} = (0^{m\ell}, 0^{k'})$. Then, we apply C_1 to the result, where we interpret a_{i-1} as a string of length ℓ over $\{0,1\}^{\frac{m\ell+k'}{\ell}}$, and consider the redundancies

$$\begin{aligned} q_{i-1} &= \mathsf{R}_{\mathsf{col}}(a_{i-1}) \in \{0,1\}^{k'} \\ r_{i-1,j} &= \mathsf{R}_{\mathsf{C}_1}\Big(\mathsf{C}_1(a_{i-1})_j\Big) \in \{0,1\}^{\rho_1 \cdot \frac{m\ell+k'}{\ell}} . \end{aligned}$$

  Lastly, the second part of $\mathsf{TC}_{\mathsf{out}}(x)_{i,j}$ is taken to be $\Big(\mathsf{spr}_{\ell}(q_{i-1})_j \, , r_{i-1,j}\Big)$.

- For the third part, we apply $g_{x,i}$ to the field element $v_j = \psi\left(\left\lfloor \frac{j}{2b} \right\rfloor\right)$. Note that this is well-defined due to the assumption $m \geq 2\log\ell$. Then, we map the evaluation to $\{0,1\}^m$ to obtain $w_j = \varphi^{-1}\left(g_{x,i}\left(v_j\right)\right)$, where w_j is reinterpreted as a string in $\left(\{0,1\}^{\frac{m}{b}}\right)^b$. Then, we take the third part of $\mathsf{TC}_{\mathsf{out}}(x)_{i,j}$ to be $\mathsf{C}_2\left(w_j\right)_{j \bmod 2b} \in \{0,1\}^{\frac{m}{b}}$.

To summarize, at time $(i,j) \in [0,\ell') \times [0,\ell)$ we output

$$\mathsf{TC}_{\mathsf{out}}(x)_{i,j} = \Big(\mathsf{TC}^{\Delta}_{\mathsf{in}}(\bar{x}_i)_j\,, \Big(\mathsf{spr}_{\ell}(q_{i-1})_j\,, r_{i-1,j}\Big)\,, \mathsf{C}_2\left(w_j\right)_{j \bmod 2b}\Big)\,.$$

Observe that by the fact that $\mathsf{TC}_{\mathsf{col}}$ is online, for every $i \in [1,\ell')\,, j \in [0,\ell)\,,\ q_{i-1}, r_{i-1,j}$ and w_j are determined by elements that lie in rows 0 to $i-1$, and in the case of $i=0$, they are set to be the respective zero strings. Thus, and since $\mathsf{TC}^{\Delta}_{\mathsf{in}}$ is online, $\mathsf{TC}_{\mathsf{out}}$ is an online function. In addition, $\mathsf{TC}_{\mathsf{out}}$ is systematic as $\mathsf{TC}^{\Delta}_{\mathsf{in}}$ is systematic. Further, since $\mathsf{TC}^{\Delta}_{\mathsf{in}}, \mathsf{TC}_{\mathsf{col}}, \mathsf{C}_1, \mathsf{C}_2, \mathsf{R}_{\mathsf{col}}$ and $\mathsf{R}_{\mathsf{C}_1}$ all are explicit, we have that $\mathsf{TC}_{\mathsf{out}}$ is explicit.

4.2.3 Distance analysis

We turn to analyze the distance pf $\mathsf{TC}_{\mathsf{out}}$. Let $x, y \in (\{0,1\}^m)^{\ell_{\mathsf{out}}}$ be distinct messages, set $s = \mathsf{split}(x,y)$, and let $d \in [0, \ell_{\mathsf{out}} - s)$. We use the $\ell' \times \ell$ row-major matrices $\mathcal{M}_x$ and $\mathcal{M}_y$ to analyze the distance. We denote the entry indices that hold the split symbols, x_s, y_s, by (s_1, s_2), and the entry indices that hold the "test" symbols, x_{s+d}, y_{s+d}, by (t_1, t_2). That is, $\mathcal{M}_x(s_1,s_2) = x_s,\ \mathcal{M}_x(t_1,t_2) = x_{s+d}$, and similarly for $\mathcal{M}_y$. We proceed by a case analysis.

The case $t_1 = s_1$.

In this case, the split index and the "test" index are in the same row. Observe that $\mathsf{TC}^{\Delta}_{\mathsf{in}}(\bar{x}_{s_1})_{[s_2,t_2]}$ and $\mathsf{TC}^{\Delta}_{\mathsf{in}}(\bar{y}_{s_1})_{[s_2,t_2]}$ are embedded in $\mathsf{TC}_{\mathsf{out}}(x)_{[s,s+d]}, \mathsf{TC}_{\mathsf{out}}(y)_{[s,s+d]}$, respectively. By Claim 4.6, $\mathsf{TC}^{\Delta}_{\mathsf{in}}$ has distance δ. Hence,

$$\mathsf{dist}\Big(\mathsf{TC}_{\mathsf{out}}(x)_{[s,s+d]}\,, \mathsf{TC}_{\mathsf{out}}(y)_{[s,s+d]}\Big) \geq \delta(d+1).$$

This Concludes the case $t_1 = s_1$. For the remaining cases, define

$$\tau = \begin{cases} t_1 - 1 & \text{if } t_2 = \ell - 1 \\ t_1 - 2 & \text{otherwise.} \end{cases}$$

Namely, τ is the maximal row index that holds a codeword symbol of $\mathsf{TC}_{\mathsf{col}}$ that is fully written. In addition, set $\sigma = \tau - s_1 + 1$. Note that σ equals to the number of codewords symbols of $\mathsf{TC}_{\mathsf{col}}$ that are fully written.

The case $t_1 = s_1 + 1$ and $\sigma = 0$.

In this case the test index is in the consecutive row to the split and $t_2 < \ell - 1$. Define $s_3 = \ell - s_2 - 1$, that is, s_3 is the column of the split when we read the row from right to left. Note that

$$g_{x,t_1}(Z) = g_{x,s_1+1}(Z) = \sum_{j=0}^{\ell-1} \varphi\left(\bar{x}_{s_1,j}\right) Z^{\ell-j-1}$$
$$g_{y,t_1}(Z) = g_{y,s_1+1}(Z) = \sum_{j=0}^{\ell-1} \varphi\left(\bar{y}_{s_1,j}\right) Z^{\ell-j-1}$$

are polynomials of degree (exactly) s_3, and recall that $\mathbb{F}$ is a field of at least ℓ elements. Hence, by a direct corollary of the fundamental theorem of algebra,

$$|\{\, e \in \mathbb{F} \mid g_{x,t_1}(e) = g_{y,t_1}(e) \,\}| \leq s_3.$$

Furthermore, since φ is bijective, there are at most s_3 field elements e for which $\varphi^{-1}\left(g_{x,t_1}(e)\right) = \varphi^{-1}\left(g_{y,t_1}(e)\right)$. Since C_2 is an error-correcting code with relative distance $\frac{1}{2}$, we get that for every $e \in \mathbb{F}$ for which $g_{x,t_1}(e) \neq g_{y,t_1}(e)$,

$$\mathsf{dist}\Big(\mathsf{C}_2\big(\varphi^{-1}\left(g_{x,t_1}(e)\right)\big), \mathsf{C}_2\big(\varphi^{-1}\left(g_{y,t_1}(e)\right)\big)\Big) \geq \frac{1}{2} \cdot 2b = b,$$

where we identify $\{0,1\}^m$ with $\left(\{0,1\}^{\frac{m}{b}}\right)^b$ in the natural way. Further observe that for every $z \in \left\{0, \dots, \left\lfloor \frac{t_2+1}{2b} \right\rfloor - 1\right\}$, the entire codewords

$$\mathsf{C}_2\big(\varphi^{-1}\left(g_{x,t_1}\left(\psi(z)\right)\right)\big), \mathsf{C}_2\big(\varphi^{-1}\left(g_{y,t_1}\left(\psi(z)\right)\right)\big)$$

are embedded in $\mathsf{TC}_{\mathsf{col}}(x)_{[s,s+d]}$ and $\mathsf{TC}_{\mathsf{col}}(y)_{[s,s+d]}$, respectively. Hence, if $s_3 + 1 \leq \frac{t_2+1}{2(\Delta+b)}$,

$$\begin{aligned}
\frac{1}{d+1} \cdot \mathsf{dist}\Big(\mathsf{TC}_{\mathsf{out}}(x)_{[s,s+d]}, \mathsf{TC}_{\mathsf{out}}(y)_{[s,s+d]}\Big) &\geq \frac{b\left(\left\lfloor \frac{t_2+1}{2b} \right\rfloor - s_3\right)}{d+1} \\
&\geq \frac{b\left(\frac{t_2+1}{2b} - 1 - s_3\right)}{s_3 + 1 + t_2 + 1} \\
&\geq \frac{b\left(\frac{t_2+1}{2b} - \frac{t_2+1}{2(\Delta+b)}\right)}{\frac{t_2+1}{2(\Delta+b)} + t_2 + 1} \\
&= \frac{\Delta}{2\Delta + 2b + 1}.
\end{aligned}$$

Otherwise, $s_3+1 > \frac{t_2+1}{2(\Delta+b)}$. Recall that $\mathsf{TC}^{\Delta}_{\mathsf{in}}$ has suffix-distance Δ and that $\mathsf{TC}^{\Delta}_{\mathsf{in}}(\bar{x}_{s_1})_{[s_2,\ell)}$ and $\mathsf{TC}^{\Delta}_{\mathsf{in}}(\bar{y}_{s_1})_{[s_2,\ell)}$ are embedded in $\mathsf{TC}_{\mathsf{out}}(x)_{[s,s+d]}$ and $\mathsf{TC}_{\mathsf{out}}(y)_{[s,s+d]}$, respectively. Hence,

$$\begin{aligned}\frac{1}{d+1}\cdot \mathsf{dist}\big(\mathsf{TC}_{\mathsf{out}}(x)_{[s,s+d]}, \mathsf{TC}_{\mathsf{out}}(y)_{[s,s+d]}\big) &\geq \frac{\Delta(\ell-s_2)}{d+1}\\ &= \frac{\Delta(s_3+1)}{s_3+1+t_2+1}\\ &> \frac{\Delta(s_3+1)}{s_3+1+2(s_3+1)(\Delta+b)}\\ &= \frac{\Delta}{2\Delta+2b+1}.\end{aligned}$$

The case $\sigma \geq 1$.

As s is the split index of x, y when interpreted as strings in $(\{0,1\}^m)^{\ell_{\mathsf{out}}}$, s_1 is their split index when interpreted as strings in $\left(\{0,1\}^{m\ell}\right)^{\ell'}$. Since $\mathsf{TC}_{\mathsf{col}}$ has relative distance $\frac{1}{8b}$ we get that

$$\mathsf{dist}\left(\mathsf{TC}_{\mathsf{col}}(x)_{[s_1,\tau]}, \mathsf{TC}_{\mathsf{col}}(y)_{[s_1,\tau]}\right) \geq \frac{1}{8b}(\tau - s_1 + 1) = \frac{1}{8b}\sigma.$$

Since C_1 has relative distance δ_1, for every $i \in [s_1, \tau]$ for which $\mathsf{TC}_{\mathsf{col}}(x)_i \neq \mathsf{TC}_{\mathsf{col}}(y)_i$ we have that

$$\mathsf{dist}\big(\mathsf{C}_1(\mathsf{TC}_{\mathsf{col}}(x))_i, \mathsf{C}_1(\mathsf{TC}_{\mathsf{col}}(y))_i\big) \geq \delta_1\ell,$$

where we interpret $\mathsf{TC}_{\mathsf{col}}(x)_i$ and $\mathsf{TC}_{\mathsf{col}}(y)_i$ as strings in $\left(\{0,1\}^{\frac{m\ell+k'}{\ell}}\right)^{\ell}$. Hence,

$$\begin{aligned}\sum_{i\in[s_1,\tau]} \mathsf{dist}\big(\mathsf{C}_1\big(\mathsf{TC}_{\mathsf{col}}(x)_i\big), \mathsf{C}_1\big(\mathsf{TC}_{\mathsf{col}}(y)_i\big)\big) &\geq \sum_{\substack{i\in[s_1,\tau] \text{ s.t.}\\ \mathsf{TC}_{\mathsf{col}}(x)_i \neq \mathsf{TC}_{\mathsf{col}}(y)_i}} \delta_1\ell\\ &\geq \frac{1}{8b}\sigma\delta_1\ell.\end{aligned}$$

Observe that $x_{[s,s+d]}$ is embedded in the first entries of $\mathsf{TC}_{\mathsf{out}}(x)_{[s,s+d]}$, and in addition, for every $i \in [s_1, \tau]$ and $j \in [0, \ell)$,

$$\left(\mathsf{spr}_{\ell}(\mathsf{R}_{\mathsf{col}}(\mathsf{TC}_{\mathsf{col}}(x)_i))_j, \mathsf{R}_{\mathsf{C}_1}\left(\mathsf{C}_1(\mathsf{TC}_{\mathsf{col}}(x)_i)_j\right)\right)$$

is embedded in $\mathsf{TC}_{\mathsf{out}}(x)_{[s,s+d]}$. Thus, we get that the spread of the entire encoding

$$\mathsf{C}_1\big(\mathsf{TC}_{\mathsf{col}}(x)_{s_1}\big), \ldots, \mathsf{C}_1(\mathsf{TC}_{\mathsf{col}}(x)_{\tau})$$

is embedded in different but fixed parts of $\mathsf{TC}_{\mathsf{out}}(x)_{[s,s+d]}$. Similarly, the entire encoding

$$\mathsf{C}_1\big(\mathsf{TC}_{\mathsf{col}}(y)_{s_1}\big), \ldots, \mathsf{C}_1\big(\mathsf{TC}_{\mathsf{col}}(y)_{\tau}\big)$$

is embedded in the corresponding parts of $\mathsf{TC}_{\mathsf{out}}(y)_{[s,s+d]}$. Therefore, using Claim 4.10, $\mathsf{TC}_{\mathsf{out}}(x)_{[s,s+d]}$ and $\mathsf{TC}_{\mathsf{out}}(y)_{[s,s+d]}$ differ in at least $\frac{1}{2} \cdot \frac{1}{8b}\sigma\delta_1\ell$ symbols. Hence,

$$\begin{aligned}
\frac{1}{d+1} \cdot \mathsf{dist}\Big(\mathsf{TC}_{\mathsf{out}}(x)_{[s,s+d]}, \mathsf{TC}_{\mathsf{out}}(y)_{[s,s+d]}\Big) &\geq \frac{1}{16b} \cdot \frac{\delta_1\sigma\ell}{d+1} \\
&\geq \frac{1}{16b} \cdot \frac{\delta_1\sigma\ell}{(\sigma+2)\ell} \\
&\geq \frac{\delta_1}{48b}
\end{aligned}$$

where the last inequality follows as $\sigma \geq 1$.

Denote the distance of $\mathsf{TC}_{\mathsf{out}}$ by δ_{out}. By the above case analysis, we conclude that

$$\delta_{\mathsf{out}} \geq \min\left\{\delta, \frac{\Delta}{2\Delta+2b+1}, \frac{\delta_1}{48b}\right\}.$$

By setting $\delta_0 = \frac{1}{11}$, $\delta_{\mathsf{min}} = \min\{\delta, \delta_0\}$ and taking

$$\Delta = \frac{2}{5}\sqrt{\delta_{\mathsf{min}}},\ b = \frac{1}{100\sqrt{\delta_{\mathsf{min}}}},\ \delta_1 = \frac{12}{25}\sqrt{\delta_{\mathsf{min}}}, \tag{4.4}$$

we get that $\Delta \in \big(0, \frac{1}{8}\big]$, $\delta_1 \leq \frac{1}{4}$ (hence $\frac{1}{m} \leq \rho_1 \leq 1$), and $\delta_{\mathsf{out}} \geq \delta_{\mathsf{min}}$ as desired.

4.2.4 Analyzing the redundancy

Let $x \in (\{0,1\}^m)^{\ell_{\mathsf{out}}}$ and let $(i,j) \in [0,\ell') \times [0,\ell)$. For $a \in \{1,2,3\}$, set $k_{\mathsf{out}}^{(a)}$ be the number of bits we output in the a^{th} entry of $\mathsf{TC}_{\mathsf{out}}(x)$ at any given time, excluding the systematic part that appears in the first entry.

- The first part of the output is $\mathsf{TC}_{\mathsf{in}}^{\Delta}(\bar{x}_i)_j$. Since $\mathsf{TC}_{\mathsf{in}}$ is an (α,β,γ) tree code, Claim 4.6 implies that $\mathsf{TC}_{\mathsf{in}}^{\Delta}$ is an $(\alpha+8\Delta,\beta,\gamma)$ tree code. Thus, for every m bits read, we output for the first part $(\alpha+8\Delta)m+\beta\ell^{\gamma}$ redundant bits.
- For the second part, we output a $\frac{1}{\ell}$ fraction of the redundant part of a symbol of $\mathsf{TC}_{\mathsf{col}}(x)_{i-1}$ concatenated with the redundant part of $\mathsf{C}_1\big(\mathsf{TC}_{\mathsf{col}}(x)_{i-1}\big)$. Since $\mathsf{TC}_{\mathsf{col}}$ is

a $\left(\frac{1}{b}, \frac{1}{b}, 1\right)$ tree code, for the second part we output

$$\begin{aligned}
k_{\mathsf{out}}^{(2)} &= \frac{k'}{\ell} + \frac{\rho_1(m\ell + k')}{\ell} \\
&= \frac{(1+\rho_1)k' + \rho_1 m\ell}{\ell} \\
&\le \frac{(1+\rho_1)\left(\frac{m\ell}{b} + \frac{\ell'}{b}\right) + \rho_1 m\ell}{\ell} \\
&= \left(\frac{1}{b} + \frac{\rho_1}{b} + \rho_1\right) m + \frac{(1+\rho_1)\ell'}{b\ell}.
\end{aligned}$$

- In the third part, for every input symbol read, we output a codeword symbol of C_2 and therefore $k_{\mathsf{out}}^{(3)} = \frac{m}{b}$.

Putting it all together, the number of redundant bits we output per (m bit) symbol read is

$$\begin{aligned}
k_{\mathsf{out}} &= k_{\mathsf{out}}^{(1)} + k_{\mathsf{out}}^{(2)} + k_{\mathsf{out}}^{(3)} \\
&\le (\alpha + 8\Delta)m + \beta\ell^{\gamma} + \left(\frac{1}{b} + \frac{\rho_1}{b} + \rho_1\right) m + \frac{(1+\rho_1)\ell'}{b\ell} + \frac{m}{b} \\
&= \left(\alpha + 8\Delta + \frac{2}{b} + \frac{\rho_1}{b} + \rho_1\right) m + \beta\ell^{\gamma} + \frac{(1+\rho_1)\ell'}{b\ell}.
\end{aligned}$$

Since $\rho_1 = 4\delta_1 = \frac{48}{25}\sqrt{\delta_{\mathsf{min}}}$, $\Delta = \frac{2}{5}\sqrt{\delta_{\mathsf{min}}}$, $b = \frac{1}{100\sqrt{\delta_{\mathsf{min}}}}$, and by setting $\ell' = \ell^{1+\gamma}$ (which implies that $\ell_{\mathsf{out}} = \ell^{2+\gamma}$), we obtain a tree code $\mathsf{TC}_{\mathsf{out}} : (\{0,1\}^m)^{\ell_{\mathsf{out}}} \to \left(\{0,1\}^m \times \{0,1\}^{k_{\mathsf{out}}}\right)^{\ell_{\mathsf{out}}}$ such that

$$\begin{aligned}
k_{\mathsf{out}} &\le \left(\alpha + 4\sqrt{\delta_{\mathsf{min}}} + 200\sqrt{\delta_{\mathsf{min}}} + 200\delta_{\mathsf{min}} + 2\sqrt{\delta_{\mathsf{min}}}\right) m + \beta\ell_{\mathsf{out}}^{\frac{\gamma}{2+\gamma}} + 100\sqrt{\delta_{\mathsf{min}}}\left(1 + 2\sqrt{\delta_{\mathsf{min}}}\right)\ell_{\mathsf{out}}^{\frac{\gamma}{2+\gamma}} \\
&= \left(\alpha + 206\sqrt{\delta_{\mathsf{min}}} + 200\delta_{\mathsf{min}}\right) m + \left(\beta + 100\sqrt{\delta_{\mathsf{min}}} + 200\delta_{\mathsf{min}}\right)\ell_{\mathsf{out}}^{\frac{\gamma}{2+\gamma}} \\
&\le \left(\alpha + 267\sqrt{\delta_{\mathsf{min}}}\right) m + \left(\beta + 161\sqrt{\delta_{\mathsf{min}}}\right)\ell_{\mathsf{out}}^{\frac{\gamma}{2+\gamma}} \qquad (4.5)
\end{aligned}$$

where Equation (4.5) holds by $\delta_{\mathsf{min}} \le \frac{1}{11}$.

To summarize, we obtain an $\left(\alpha + O\left(\sqrt{\delta_{\mathsf{min}}}\right), \beta + O\left(\sqrt{\delta_{\mathsf{min}}}\right), \frac{\gamma}{2+\gamma}\right)$ tree code with distance $\delta_{\mathsf{out}} \ge \delta_{\mathsf{min}}$, arity 2^m, and depth $\ell_{\mathsf{out}} = \ell^{2+\gamma}$. $\square$

5 Proof of main result

Equipped with our transformation as given in Section 4, we are ready to prove Theorem 1.2. Its worth noting that we take a bottom up approach in contrast to the top down point of view that was taken in the proof overview (see Section 2).

Proof of Theorem 1.2. We first note that we might as well assume that $\delta \geq \frac{1}{(\log \log n)^2}$ as for that distance, the theorem already guarantees a tree code with a constant number of colors. Let $n \in \mathbb{N}$ and set δ_0 to be the constant from Theorem 4.11. Let $\delta \in (0, \delta_0]$, $m = \max\left\{4 \log n, \frac{2}{\delta} \log \frac{1}{\delta}\right\}$, and for every $i \in \mathbb{N}$ define $\ell_i = 2^{2^{i+1}-1}$. Set $r = \lceil \log \log n \rceil$ where, recall, all logarithms are taken base 2. We define a sequence of functions

$$\mathsf{TC}^{(i)} : (\{0,1\}^m)^{\ell_i} \to \left(\{0,1\}^m \times \{0,1\}^{k_i}\right)^{\ell_i}$$

for $i = 0, \ldots, r$ where $k_0, \ldots, k_r$ will be determined later on.

The first function, corresponding to $i = 0$, is defined to be the identity map. More precisely, we set $k_0 = 0$ and define $\mathsf{TC}^{(0)} : (\{0,1\}^m)^{\ell_0} \to (\{0,1\}^m)^{\ell_0}$ (where we identify $\{0,1\}^m \times \{0,1\}^0$ with $\{0,1\}^m$ in the natural way). Note that, as $\ell_0 = 2$, $\mathsf{TC}^{(0)}$ is a tree code with distance $\frac{1}{2}$. With our notation, $\mathsf{TC}^{(0)}$ is a $(0,0,0)$ tree code, in particular, $\mathsf{TC}^{(0)}$ is an $(\alpha_0, \beta_0, \gamma_0)$ tree code with $\alpha_0 = 0$, $\beta_0 = \gamma_0 = 1$.

For $i \geq 0$ we define $\mathsf{TC}^{(i+1)}$ to be the $(\alpha_{i+1}, \beta_{i+1}, \gamma_{i+1})$ tree code obtained by the transformation that is given by Theorem 4.11, set with distance δ, when applied to the $(\alpha_i, \beta_i, \gamma_i)$ tree code $\mathsf{TC}^{(i)}$. As $m \geq \frac{2}{\delta} \log \frac{1}{\delta}$ by definition and since the transformation preserves the distance, to verify that the hypothesis of Theorem 4.11 holds, namely, $m \geq \max\left\{2 \log \ell_i, \frac{2}{\delta} \log \frac{1}{\delta}\right\}$, it suffices to show that $m \geq 2 \log \ell_i$ for $i = 0, \ldots, r-1$. As the sequence $(\ell_i)_i$ increases with i, it suffices to consider $i = r-1$. One can easily verify that $\ell_{r-1} \leq n^2$ and by our choice of m, we have that $m \geq 4 \log n \geq 2 \log \ell_{r-1}$ as required.

Note that, by the way the transformation is defined, the sequences $(\ell_i)_i, (\gamma_i)_i$ satisfy the recursive relations

$$\begin{aligned} \ell_{i+1} &= \ell_i^{2+\gamma_i}, \\ \gamma_{i+1} &= \frac{\gamma_i}{2+\gamma_i}. \end{aligned}$$

We wish to verify that this agrees with our definition of the sequence $(\ell_i)_i$. Indeed, a simple induction can be used to show that $\gamma_i = \frac{1}{2^{i+1}-1}$ where we use the fact that $\gamma_0 = 1$.

As we defined $\ell_i = 2^{2^{i+1}-1}$, we have that

$$\ell_i^{2+\gamma_i} = \left(2^{2^{i+1}-1}\right)^{2+\gamma_i} = \left(2^{2^{i+1}-1}\right)^{\frac{2^{i+2}-1}{2^{i+1}-1}} = 2^{2^{i+2}-1} = \ell_{i+1}.$$

We turn to bound the size of the output alphabet of $\mathsf{TC}^{(r)}$. Using Theorem 4.11, a straightforward inductive argument shows that $\alpha_i, \beta_i = O(i\sqrt{\delta})$. After applying the transformation for r iterations, we get

$$\mathsf{TC}^{(r)} : (\{0,1\}^m)^{\ell_r} \to \left(\{0,1\}^m \times \{0,1\}^{k_r}\right)^{\ell_r}$$

where $m + k_r = (1+\alpha_r)m + \beta_r \ell_r^{\gamma_r}$. Note that $\ell_r^{\gamma_r} = 2$ (which formalizes the key idea of eliminating the dependence on the tree's depth altogether) and so

$$m + k_r \le (1 + O(\sqrt{\delta} \cdot \log\log n))m. \tag{5.1}$$

By a careful inspection, $\mathsf{TC}^{(0)}$ is indeed explicit and since the transformation applied to each $\mathsf{TC}^{(i)}$ is efficient (in ℓ_i, and so in particular in n), and since we apply the transformation for $r = O(\log\log n)$ iterations, we have that $\mathsf{TC}^{(r)}$ is explicit.

We define

$$\mathsf{TC}' : (\{0,1\}^m)^{\frac{n}{m}} \to (\{0,1\}^{m+k_r})^{\frac{n}{m}}$$

to be the truncation of $\mathsf{TC}^{(r)}$ to the length $\frac{n}{m}$ prefix. Indeed, $\frac{n}{m} \le \ell_r$ as can be easily verified. In addition, by the properties of $\mathsf{TC}^{(r)}$, TC' is explicit and has distance δ. Finally, we define the binary tree code that is asserted by the theorem to be the binary tree code with distance $c_{\mathsf{dist}} \cdot \delta$ that is given by Fact 3.11 when fed with TC' as input, where $c_{\mathsf{rate}}, c_{\mathsf{dist}}$ are the constants from the statement of Fact 3.11. Namely,

$$\mathsf{TC}_{\mathsf{bin}} : \{0,1\}^n \to \left(\{0,1\}^{c_{\mathsf{rate}}\frac{m+k_r}{m}}\right)^n.$$

By Fact 3.11, the running time required for evaluating $\mathsf{TC}_{\mathsf{bin}}$ is $\mathrm{poly}(n, 2^m) = \mathrm{poly}(n)$. To conclude the proof, we bound the number of colors used by $\mathsf{TC}_{\mathsf{bin}}$:

$$2^{c_{\mathsf{rate}}\frac{m+k_r}{m}} = 2^{O(1+\sqrt{\delta}\log\log n)} = 2^{O(\sqrt{\delta}\log\log n)} = (\log n)^{O(\sqrt{\delta})},$$

where the first equality follows by Equation (5.1), and the second holds per our assumption $\delta \ge \frac{1}{(\log\log n)^2}$. $\square$

References

[BH20] Siddharth Bhandari and Prahladh Harsha. A note on the explicit constructions of tree codes over polylogarithmic-sized alphabet. *arXiv preprint arXiv:2002.08231*, 2020.

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

[BYCN20] Inbar Ben Yaacov, Gil Cohen, and Anand Kumar Narayanan. Candidate tree codes via pascal determinant cubes. In *Electron. Colloquium Comput. Complex.*, volume 27, page 141, 2020.

[CHS18] Gil Cohen, Bernhard Haeupler, and Leonard J. Schulman. Explicit binary tree codes with polylogarithmic size alphabet. In *STOC'18—Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 535–544. ACM, New York, 2018.

[CS20] Gil Cohen and Shahar Samocha. Palette-alternating tree codes. In *The 35th Computational Complexity Conference (CCC 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[Gel17] Ran Gelles. Coding for interactive communication: a survey. *Found. Trends Theor. Comput. Sci.*, 13(1-2):front matter, 1–157, 2017.

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

[Ham50] Richard W. Hamming. Error detecting and error correcting codes. *Bell System Tech. J.*, 29:147–160, 1950.

[Jus72] Jorn Justesen. Class of constructive asymptotically good algebraic codes. *IEEE Transactions on Information Theory*, 18(5):652–656, 1972.

[MS14] Cristopher Moore and Leonard J. Schulman. Tree codes and a conjecture on exponential sums. In *ITCS'14—Proceedings of the 2014 Conference on Innovations in Theoretical Computer Science*, pages 145–153. ACM, New York, 2014.

[NW20] Anand Kumar Narayanan and Matthew Weidner. On decoding Cohen-Haeupler-Schulman tree codes. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1337–1356. SIAM, 2020.

[Pud16] Pavel Pudlák. Linear tree codes and the problem of explicit constructions. *Linear Algebra Appl.*, 490:124–144, 2016.

[Sch93] Leonard J. Schulman. Deterministic coding for interactive communication. In *Proceedings of the 25th annual ACM Symposium on Theory of Computing*, pages 747–756, 1993.

[Sch94] Leonard J. Schulman. Postscript of 21 september 2003 to coding for interactive communication. http://users.cms.caltech.edu/ schulman/Papers/intercodingpostscript.txt, 1994.

[Sch96] Leonard J. Schulman. Coding for interactive communication. *IEEE Trans. Inform. Theory*, 42(6, part 1):1745–1756, 1996. Codes and complexity.

[Sha48] Claude E. Shannon. A mathematical theory of communication. *Bell System Tech. J.*, 27:379–423, 623–656, 1948.

[SS96] Michael Sipser and Daniel A. Spielman. Expander codes. *IEEE Transactions on Information Theory*, 42(6):1710–1722, 1996.

[TVZ82] Michael A. Tsfasman, Serge G. Vladut, and Thomas Zink. Modular curves, Shimura curves, and Goppa codes, better than Varshamov-Gilbert bound. *Mathematische Nachrichten*, 109(1):21–28, 1982.

A From finite to infinite tree codes

In this section we show how to efficiently transform a family of finite tree codes to one infinite tree code with comparable parameters.

Lemma A.1. *Assume that for every $i \in \mathbb{N}$, there exists a finite tree code $\mathsf{TC}_{2^i} : \Sigma_{\mathsf{in}}^{2^i} \to \Sigma_{\mathsf{out}}^{2^i}$ with distance δ. Then, there exists an infinite tree code $\mathsf{TC} : \Sigma_{\mathsf{in}}^{\mathbb{N}} \to (\Sigma_{\mathsf{out}}^3)^{\mathbb{N}}$ with distance $\frac{\delta}{2}$. Furthermore, given that the tree codes $(\mathsf{TC}_{2^i})_{i \in \mathbb{N}}$ are explicit, then so is TC.*

Proof. We start the proof by a somewhat informal description, for ease of readability, and then formalize the argument. We describe how the infinite tree code TC is constructed using $(\mathsf{TC}_{2^i})_{i\in\mathbb{N}}$. The idea behind the construction is that at each time we encode the input symbol using two tree codes, which can be denoted by $\mathsf{TC}_{\mathsf{current}}$ and $\mathsf{TC}_{\mathsf{next}}$. The depth of $\mathsf{TC}_{\mathsf{current}}$ is 2^i, for some i, and the depth of $\mathsf{TC}_{\mathsf{next}}$ is always double that of $\mathsf{TC}_{\mathsf{current}}$. We will use $\mathsf{TC}_{\mathsf{current}}$ and $\mathsf{TC}_{\mathsf{next}}$ for a period of length 2^{i-1} starting from time 2^{i-1}, after which they will be switched to the next pair of tree codes. For every j in that period, we output 3 symbols: one symbol that corresponds to the output of $\mathsf{TC}_{\mathsf{current}}$ on $x_{[0,2^i)}$ at time j, and two symbols of $\mathsf{TC}_{\mathsf{next}}$. The two symbols of $\mathsf{TC}_{\mathsf{next}}$ that we output correspond to simulating $\mathsf{TC}_{\mathsf{next}}$ on the entire $x_{[0,2^i)}$. Note that after a period of length 2^{i-1}, the entire output of $\mathsf{TC}_{\mathsf{next}}$ on $x_{[0,2^i)}$ is written. After this period is over, $\mathsf{TC}_{\mathsf{next}}$ is set to be $\mathsf{TC}_{\mathsf{current}}$ and $\mathsf{TC}_{\mathsf{next}}$ is set to be a tree code of a double length, and the process infinitely continues in this manner.

We turn to formally describe the construction and its analysis. Let $\perp$ be any fixed symbol of Σ_{out}. On input $x \in \Sigma_{\mathsf{in}}^{\mathbb{N}}$, at time $j = 0$, the j^{th} output is defined by

$$\mathsf{TC}(x)_0 = \left(\mathsf{TC}_1\left(x_{[0,0]}\right)_0, \mathsf{TC}_2\left(x_{[0,1]}\right)_0, \perp\right).$$

At time $j > 0$, the j^{th} output is

$$\begin{aligned}\mathsf{TC}(x)_j = \Bigg(&\mathsf{TC}_{2^{\lceil\log(j+1)\rceil}}\left(x_{\left[0,2^{\lceil\log(j+1)\rceil}\right)}\right)_j,\\ &\mathsf{TC}_{2^{\lceil\log(j+1)\rceil+1}}\left(x_{\left[0,2^{\lceil\log(j+1)\rceil+1}\right)}\right)_{2\left(j-2^{\lceil\log(j+1)\rceil-1}\right)},\\ &\mathsf{TC}_{2^{\lceil\log(j+1)\rceil+1}}\left(x_{\left[0,2^{\lceil\log(j+1)\rceil+1}\right)}\right)_{2\left(j-2^{\lceil\log(j+1)\rceil-1}\right)+1}\Bigg).\end{aligned}$$

We turn to analyze the properties of TC. First note that TC is indeed online, which follows from that every TC_{2^i} is online, and for every j, $\mathsf{TC}(x)_j$ only depends on values $\mathsf{TC}_{2^i}\left(x_{[0,2^i)}\right)_{j'}$ for $j' \le j$. Further note that indeed the output alphabet of TC is Σ_{out}^3. Thirdly, if $(\mathsf{TC}_{2^i})_i$ are explicit, then by the definition of $\mathsf{TC}(x)_j$, the time to compute $\mathsf{TC}(x)_j$ is

$$\mathrm{poly}\left(\log|\Sigma_{\mathsf{in}}|, 2^{\lceil\log(j+1)\rceil+1}\right) = \mathrm{poly}\left(\log|\Sigma_{\mathsf{in}}|, j+1\right),$$

and so TC is explicit.

Lastly, we argue that TC is a tree code with distance at least $\frac{\delta}{2}$. Given any two inputs $x, x' \in \Sigma_{\mathsf{in}}^{\mathbb{N}}$, $x \neq x'$ such that $s = \mathsf{split}(x,x')$ and $\ell \in \mathbb{N}$, we need to show that

$$\mathsf{dist}\Big(\mathsf{TC}(x)_{[s,s+\ell]}, \mathsf{TC}(x')_{[s,s+\ell]}\Big) \ge \frac{\delta}{2}(\ell+1). \tag{A.1}$$

By assumption, for $i = \lceil \log(s+\ell+1) \rceil$, we have that

$$\mathsf{dist}\Big(\mathsf{TC}_{2^i}\left(x_{[0,2^i)}\right)_{[s,s+\ell]}, \mathsf{TC}_{2^i}\left({x'}_{[0,2^i)}\right)_{[s,s+\ell]}\Big) \geq \delta(\ell+1),$$

and by construction, the elements of $\mathsf{TC}_{2^i}\left(x_{[0,2^i)}\right)_{[s,s+\ell]}$ are embedded in $\mathsf{TC}(x)_{[s,s+\ell]}$, where each element of $\mathsf{TC}(x)_{[s,s+\ell]}$ contains at most two elements of $\mathsf{TC}_{2^i}\left(x_{[0,2^i)}\right)_{[s,s+\ell]}$. The same holds for $\mathsf{TC}_{2^i}\left({x'}_{[0,2^i)}\right)_{[s,s+\ell]}$ and $\mathsf{TC}(x')_{[s,s+\ell]}$. On account of this, Equation (A.1) immediately follows, as required. □

B The CHS constructions

In [CHS18], the authors present a construction of an explicit, systematic tree code over the integers. Here we present a variation of their construction that has a small dependence on α.

B.1 The CHS tree code over the integers

Theorem B.1 ([CHS18] Theorem 1.3)**.** *There exists an explicit and systematic tree code $\mathsf{TC}_{\mathbb{Z}} : \mathbb{Z}^{\mathbb{N}} \to (\mathbb{Z}\times\mathbb{Z})^{\mathbb{N}}$ with distance $\frac{1}{2}$. Moreover, for every $z \in \mathbb{Z}^{\mathbb{N}}$ and $t \in \mathbb{N}$, $\left|(\mathsf{TC}_{\mathbb{Z}}(z)_t)_1\right| \leq 2^t \cdot \max\{|z_0|, \dots, |z_t|\}$.*

We give a sketch of their construction and analysis. To construct their tree code, [CHS18] make use of the Newton basis.

Definition B.2 (The Newton basis)**.** *For every $k \in \mathbb{N}$, define the univariate real polynomial $\binom{X}{k} \in \mathbb{R}[X]$ by*

$$\binom{X}{k} = \frac{1}{k!}\prod_{i=0}^{k-1} X - i.$$

Observe that for every $d \in \mathbb{N}$, the set $\left\{ \binom{X}{k} \;\middle|\; k \in [0,d] \right\}$ forms a basis for the space of univariate real polynomials of degree at most d. This basis is called the Newton basis.

[CHS18]'s construction of $\mathsf{TC}_{\mathbb{Z}}$ is as follows. Given a message $z \in \mathbb{Z}^{\mathbb{N}}$ and $t \in \mathbb{N}$, set $f^{(t)} : \mathbb{N} \to \mathbb{Z}$ to be the polynomial of least degree such that for every $i \in [0,t]$, $f^{(t)}(i) = z_i$. By [CHS18] Lemma 3.8, one can expand $f^{(t)}$ in the Newton basis to obtain

$$f^{(t)}(T) = \sum_{i=0}^{t} \lambda_i \binom{T}{i}$$

where for every $i \in [0,t]$, $\lambda_i \in \mathbb{Z}$ and is determined by $z_0, \dots, z_i$. Then, the resulted tree code is defined by $\mathsf{TC}_{\mathbb{Z}}(z)_t = (z_t, \lambda_t)$, $t \in \mathbb{N}$. Note that for every $t \in \mathbb{N}$,

$$f^{(t)}(T) = f^{(t-1)}(T) + \lambda_t \binom{T}{t}$$

and therefore $\mathsf{TC}_{\mathbb{Z}}$ is well defined. The authors prove that $\mathsf{TC}_{\mathbb{Z}}$ has distance $\frac{1}{2}$ using their sparsity lemma.

Lemma B.3 (The sparsity lemma, [CHS18] Lemma 1.4)**.** *Let $f \in \mathbb{R}[X]$ be a nonzero polynomial of sparsity $s \geq 1$, expanded in the Newton basis. Let $a \in \mathbb{N}$ be the least integer for which $f(a) \neq 0$. Then, f has at most $s-1$ distinct roots in $[a, \infty) \cap \mathbb{Z}$.*

B.2 CHS's tree code with smaller number of colors

We use the infinite tree code from Theorem B.1 to deduce an explicit finite tree code over a finite alphabet. The following corollary is similar to Corollary 6.1 in [CHS18], only that here m and ℓ are decoupled.

Corollary B.4. *For every positive integers m, ℓ, there exists an explicit and systematic tree code*

$$\mathsf{TC}_{m,\ell} : (\{0,1\}^m)^\ell \to \left(\{0,1\}^m \times \{0,1\}^{m+\ell}\right)^\ell$$

with distance $\frac{1}{2}$.

For our transformation, we make use of Claim 4.2 that generalizes Corollary B.4. We give a proof sketch for Corollary B.4 for completeness and provide a formal proof for Claim 4.2 in Section 4.

Proof sketch. Let $\mathsf{TC}_{\mathbb{Z}} : \mathbb{Z}^{\mathbb{N}} \to (\mathbb{Z} \times \mathbb{Z})^{\mathbb{N}}$ be the tree code given by Theorem B.1 and let $x = (x_0, \dots, x_{\ell-1}) \in (\{0,1\}^m)^\ell$ be a message to be encoded. We define

$$f(T) = \sum_{i=0}^{\ell-1} \lambda_i \binom{T}{i}$$

to be the polynomial of least degree such that for every $t \in [0, \ell)$, $f(t) = x_t$, where we identify $\{0,1\}^m$ with $\{0, \dots, 2^m - 1\}$ in an arbitrary but fixed way. By Lemma 3.8 in [CHS18], $f(T) \in \mathbb{Z}[T]$ and for every $t \in [0, \ell)$, λ_t is determined by $x_0, \dots, x_t$. Then, we define

$$\mathsf{TC}_{m,\ell}(x)_t = (x_t, \lambda_t)\,.$$

For analyzing the output alphabet size, let $t \in [0, \ell)$. Note that $|x_t| \leq 2^{m-1} - 1$ when represented as an integer. By [CHS18] Lemma 3.8,

$$\lambda_t = \sum_{i=0}^{t} (-1)^{t-i} \binom{t}{i} x_i$$

and therefore

$$|\lambda_t| \leq \max_{i \in [0,t]} \{|x_i|\} \sum_{i=0}^{t} \binom{t}{i} \leq \left(2^{m-1} - 1\right) 2^t \leq 2^{m+\ell-1} - 1. \tag{B.1}$$

Thus, we can represent $\mathsf{TC}_{m,\ell}(x)_t$ by $2m + \ell$ bits (including the sign of λ_t, which can be encoded in an arbitrary but fixed way). Moreover, observe that one can think of $\mathsf{TC}_{m,\ell}$ as a truncated version of $\mathsf{TC}_{\mathbb{Z}}$ over a subset of its input alphabet. Therefore, by Lemma B.3 and by a very similar analysis to the one provided for concluding the distance of $\mathsf{TC}_{\mathbb{Z}}$, $\mathsf{TC}_{m,\ell}$ has distance $\frac{1}{2}$. □

In our notations, $\mathsf{TC}_{m,\ell}$ is a $(1, 1, 1)$ tree code. The only drawback with using this construction for our aims is that the dependence on m (namely, α) is too large for our needs. This prevents our transformation from achieving the sublogarithmic alphabet size. Thus, we need to reduce the dependence on m while not deteriorating the other parameters too much. For this task, we make use of a variation of the above construction which is a $\left(\frac{1}{b}, \frac{1}{b}, 1\right)$ tree code for a positive integer b, in which the choice of b affects the resulted distance.

Claim B.5 (Claim 4.2, restated)**.** *For every positive integers b, c, m, ℓ such that $b \leq c \leq 2^{\frac{m+\ell}{b}}$, there exists an explicit and systematic tree code*

$$\mathsf{TC}^{b,c}_{m,\ell} : \left(\{0,1\}^m\right)^\ell \to \left(\{0,1\}^m \times \{0,1\}^{\frac{m+\ell}{b}}\right)^\ell$$

with distance $\frac{c-b}{2c^2}$.

Proof. Let b, c, m, ℓ be positive integers that satisfy the above constraints and set

$$\mathsf{EncRS} : \left(\{0,1\}^{\frac{m+\ell}{b}}\right)^b \to \left(\{0,1\}^{\frac{m+\ell}{b}}\right)^c$$

to be the error-correcting code provided by Definition 3.6. It follows from Fact 3.7 that

EncRS has relative distance $\geq 1 - \frac{b}{c}$. Given a string $x = (x_0, \dots, x_{\ell-1}) \in (\{0,1\}^m)^\ell$, set

$$f_x(T) = \sum_{i=0}^{\ell-1} \lambda_i \binom{T}{i}$$

to be the polynomial of least degree such that for every $t \in [0, \ell)$, $f_x(t) = x_t$, where we identify $\{0,1\}^m$ with $\{0, \dots, 2^m - 1\}$. By [CHS18], Lemma 3.8, for every $t \in [0, \ell)$, λ_t is an integer that is determined by $x_0, \dots, x_t$. At time $t \in [0, \ell)$, define the output of $\mathsf{TC}^{b,c}_{m,\ell}$ by

$$\mathsf{TC}^{b,c}_{m,\ell}(x)_t = \left(x_t, \mathsf{EncRS}\left(\lambda_{\lfloor \frac{t}{c} \rfloor c}\right)_{t \bmod c}\right)$$

where λ_t is interpreted as a string of length b over $\{0,1\}^{\frac{m+\ell}{b}}$ (this can be done by Equation (B.1)). Observe that we can think of $\mathsf{TC}^{b,c}_{m,\ell}$ as a punctured version of the tree code $\mathsf{TC}_{m,\ell}$ from Corollary B.4, such that we calculate a coefficient once per every c symbols read, encode it, and spread the result over c output symbols. Hence, $\mathsf{TC}^{b,c}_{m,\ell}$ is online, and by the explicitness of $\mathsf{TC}_{m,\ell}$ and EncRS, $\mathsf{TC}^{b,c}_{m,\ell}$ is explicit.

As for the redundancy, note that for every $t \in [0, \ell)$ one can represent $\mathsf{TC}^{b,c}_{m,\ell}(x)_t$ by $m + \frac{m+\ell}{b}$ bits. For analyzing the distance, let $x, y \in (\{0,1\}^m)^\ell$ be distinct strings, set

$$f_x(T) = \sum_{i=0}^{\ell-1} \lambda_i \binom{T}{i}, \quad f_y(T) = \sum_{i=0}^{\ell-1} \mu_i \binom{T}{i}$$

to be their respective polynomials over $\mathbb{Z}$, and define $g(T) = f_x(T) - f_y(T) \in \mathbb{Z}[T]$. Set $a = \mathsf{split}(x,y)$, let $d \in [0, \ell - a)$, and define $I = [0, a+d]$. Consider two cases. If $d \geq 2c$, set

$$g^{(a+d-1)}(T) = \sum_{i=0}^{a+d-1} \nu_i \binom{T}{i}$$

to be the polynomial of least degree such that for every $t \in [0, a+d)$, $g^{(a+d-1)}(t) = x_t - y_t$. Set s to be the sparsity of $g^{(a+d-1)}$. By [CHS18], Lemma 3.8, for every $t \in I$ the t^{th} coefficient of $g^{(a+d-1)}$ and g only depends on $g^{(a+d-1)}(0), \dots, g^{(a+d-1)}(t)$ and $g(0), \dots, g(t)$ respectively. Moreover, since for every $t \in I$, $g^{(a+d-1)}(t) = g(t)$, we can bound the number of zeros of $g^{(a+d-1)}$ in I and conclude a bound for g. In particular, we have that $\nu_t = \lambda_t - \mu_t$ and so $g^{(a+d-1)}(T) \in \mathbb{Z}[T]$.

Now, since $x \neq y$, we have that $g^{(a+d-1)}$ is a nonzero polynomial. Thus, by [CHS18] Lemma 1.4, it has at most $s-1$ distinct roots in $[a, \infty) \cap \mathbb{Z}$ and therefore $\mathsf{TC}^{b,c}_{m,\ell}(x)_{[a,a+d]}$ and $\mathsf{TC}^{b,c}_{m,\ell}(y)_{[a,a+d]}$ differ in at least $d + 1 - (s-1)$ symbols, when considering only their first entries (i.e., the entries that hold the message symbols). Define

$$\mathfrak{I} = \left\{ t \in I \;\middle|\; \lambda_t \neq \mu_t \text{ and } \mathsf{EncRS}(\lambda_t) \text{ is fully embedded in } \mathsf{TC}^{b,c}_{m,\ell}(x)_{[a,a+d]} \right\}.$$

Observe that

$$|\mathfrak{I}| \geq s - 2 - \frac{c-1}{c}(d+1).$$

This bound holds since at most two coefficients may appear only partially in I, and since we output one coefficient for every c symbols read. Set $\varepsilon = 1 - \frac{b}{c}$. By Definition 3.6, for every $t \in \mathfrak{I}$, $\mathsf{EncRS}(\lambda_t)$ and $\mathsf{EncRS}(\mu_t)$ differ in at least εc symbols. Hence, the number of entries in which $\mathsf{TC}^{b,c}_{m,\ell}(x)_{[a,a+d]}$ and $\mathsf{TC}^{b,c}_{m,\ell}(y)_{[a,a+d]}$ differ as a tuple is at least

$$\max\left\{ d+1-(s-1), \varepsilon c\left(s-2-\frac{c-1}{c}(d+1)\right)\right\}.$$

If the first factor dominates, then

$$d+1-(s-1) \geq \varepsilon c\left(s-2-\frac{(c-1)(d+1)}{c}\right),$$

and therefore

$$\frac{d+\varepsilon(c-1)(d+1)+2\varepsilon c+2}{\varepsilon c+1} \geq s.$$

Hence

$$\begin{aligned} d+1-(s-1) &\geq d+2-\frac{d+\varepsilon(c-1)(d+1)+2\varepsilon c+2}{\varepsilon c+1} \\ &= \frac{\varepsilon}{\varepsilon c+1}(d-c+1) \\ &\geq \frac{\varepsilon}{\varepsilon c+1}\left(\frac{d}{2}+1\right) && \text{(B.2)} \\ &\geq \frac{\varepsilon}{2c}(d+1) && \text{(B.3)} \end{aligned}$$

where Equation (B.2) is due to $d \geq 2c$ and Equation (B.3) is due to $c \geq \frac{1}{1-\varepsilon}$. Otherwise,

$$\frac{d+\varepsilon(c-1)(d+1)+2\varepsilon c+2}{\varepsilon c+1} < s,$$

hence,

$$\varepsilon c \left(s - 2 - \frac{(c-1)(d+1)}{c} \right) \geq \frac{\varepsilon}{2(\varepsilon c + 1)} (d+1)\,,$$

and therefore, in case of $d \geq 2c$,

$$\mathsf{dist}\Big(\mathsf{TC}^{b,c}_{m,\ell}(x)_{[a,a+d]}\,, \mathsf{TC}^{b,c}_{m,\ell}(y)_{[a,a+d]}\Big) \geq \frac{\varepsilon}{2c}(d+1).$$

Otherwise, $d < 2c$. By definition, $x_a \neq y_a$ and therefore $\mathsf{TC}^{b,c}_{m,\ell}(x)_a \neq \mathsf{TC}^{b,c}_{m,\ell}(y)_a$ as a tuple. Hence, the fraction of symbols for which $\mathsf{TC}^{b,c}_{m,\ell}(x)_{[a,a+d]}$ and $\mathsf{TC}^{b,c}_{m,\ell}(y)_{[a,a+d]}$ differ is at least

$$\frac{1}{d+1} \geq \frac{1}{2c}.$$

Therefore

$$\mathsf{dist}\Big(\mathsf{TC}^{b,c}_{m,\ell}(x)\,, \mathsf{TC}^{b,c}_{m,\ell}(y)\Big) \geq \frac{\varepsilon}{2c}(d+1).$$

Hence, $\mathsf{TC}^{b,c}_{m,\ell}$ has distance at lest

$$\frac{\varepsilon}{2c} = \frac{1-\frac{b}{c}}{2c} = \frac{c-b}{2c^2}.$$

□

C Proof of Lemma 3.8

Proof. Set $n = (1+\tau)k$. Let $\mathsf{EncRS} : (\{0,1\}^m)^k \to (\{0,1\}^m)^n$ be the systematic encoding of Reed-Solomon with respect to m, k, n and set $\mathsf{RedRS} : (\{0,1\}^m)^k \to (\{0,1\}^m)^{n-k}$ to be the Reed-Solomon redundancy function (see Definition 3.6). We define the encoding $\mathsf{C} : (\{0,1\}^m)^k \to (\{0,1\}^m \times \{0,1\}^{\tau m})^k$ as follows. For every given message $x \in (\{0,1\}^m)^k$ and every $t \in [0,k)$, define

$$\mathsf{C}(x)_t = (x_t, \mathsf{spr}_k(\mathsf{RedRS}(x))_t)\,.$$

Note that the length of $\mathsf{C}(x)_t$ in bits is

$$m + \frac{m(n-k)}{k} = (1+\tau)m.$$

We turn to analyze the distance. Let $x \neq y \in (\{0,1\}^m)^k$ and set $d = \mathsf{dist}(x,y)$. By Fact 3.7,

$$\mathsf{dist}(\mathsf{EncRS}(x), \mathsf{EncRS}(y)) \geq n - k + 1$$

and thus

$$\mathsf{dist}(\mathsf{RedRS}(x), \mathsf{RedRS}(y)) \geq n - k + 1 - d.$$

Further, since $\frac{1}{m} \leq \tau \leq 1$, we have that $n-k \leq k \leq m(n-k)$ and therefore, by Claim 4.9,

$$\mathsf{dist}(\mathsf{spr}_k(\mathsf{RedRS}(x)), \mathsf{spr}_k(\mathsf{RedRS}(y))) \geq \frac{n-k+1-d}{2}.$$

Thus,

$$\mathsf{dist}(\mathsf{C}(x), \mathsf{C}(y)) \geq \max\left\{d, \frac{n-k+1-d}{2}\right\} \geq \frac{n-k+1}{4} \geq \frac{\tau k}{4}.$$

□

D Online codes with suffix distance

Claim D.1. *For every $m, n \in \mathbb{N}$ such that $1 \leq n \leq 2^m$, $m \geq 1$, and $\Delta \in \left(0, \frac{1}{8}\right]$, there exists an online function $f : (\{0,1\}^m)^n \to \left(\{0,1\}^m \times \{0,1\}^{8\Delta m}\right)^n$ with suffix-distance Δ.*

Proof. Before we explicitly describe an encoding f that meets the requirements we set up some notations. We set

$$k = \lfloor \log(n+1) \rfloor$$

and

$$r = n - 2^k + 1. \tag{D.1}$$

Note that

$$r \leq \frac{n}{2} \leq 2^k - 1. \tag{D.2}$$

For every $x \in (\{0,1\}^m)^n$ we set $k_0 = r$ and $n_0 = r + 8\Delta 2^{k-1}$, and define

$$z_0(x) = \mathsf{RedRS}_{(m,k_0,n_0)}\left(x_{[0,r)}\right) \in (\{0,1\}^m)^{8\Delta 2^{k-1}},$$

where RedRS is as defined in Definition 3.6. Further define

$$y_0(x) = \mathsf{spr}_{2^{k-1}}(z_0(x)) \in \left(\{0,1\}^{8\Delta m}\right)^{2^{k-1}}.$$

For every $i \in [k]$ we set $k_i = 2^{k-i}$, $n_i = (1+4\Delta)2^{k-i}$, and define

$$z_i(x) = \mathsf{RedRS}_{(m,k_i,n_i)}\left(x_{\left[r+2^k-2^{k-i+1},r+2^k-2^{k-i}\right)}\right) \in \left(\{0,1\}^m\right)^{4\Delta 2^{k-i}},$$

where $\mathsf{RedRS}_{(m,k_i,n_i)}$ is as defined in Definition 3.6, and define

$$y_i(x) = \mathsf{spr}_{2^{k-i-1}}(z_i(x)) \in \left(\{0,1\}^{8\Delta m}\right)^{2^{k-i-1}}.$$

In words, for each $i \in [k]$, $y_i(x)$ is the result of the (m,k_i,n_i)-Reed-Solomon redundancy function applied to $x_{[r+2^k-2^{k-i+1},r+2^k-2^{k-i})}$, resized from a length of $n_i - k_i = 4\Delta 2^{k-i}$ to a string of length 2^{k-i-1}.

With the definitions set up, we can now describe the output of f on each input $x \in (\{0,1\}^m)^n$. For each time $t \in [0,r)$, we set the t^{th} output to be

$$f(x)_t = (x_t, \bar{0}).$$

For each time $t \in [r,n)$, it can be verified by inspection that there is exactly one pair $i \in [k]$, $j \in [2^{k-i}]$ such that $t = r + 2^k - 2^{k-i+1} + j - 1$, and let i,j be that pair. We set the t^{th} output symbol to be

$$f(x)_t = \left(x_t, y_{i-1}(x)_{j-1}\right).$$

To see that the described encoding f meets the requirements, we first note that it is online. Indeed, for $t \in [0,r)$, the t^{th} output symbol of $f(x)$ is x_t paired with $\bar{0}$. For $t \in [r,n)$, $t = r + 2^k - 2^{k-i+1} + j - 1$, the t^{th} output symbol of $f(x)$ is $\left(x_t, y_{i-1}(x)_{j-1}\right)$, and $y_{i-1}(x)$ is determined by $z_{i-1}(x)$, which is in turn determined by $x_{[0,r)}$, in the that case $i = 1$, and indeed $r - 1 < t$. In the case that $i > 1$, $z_{i-1}(x)$ is determined by $x_{[r+2^k-2^{k-i+2},r+2^k-2^{k-i+1})}$ and we have that

$$r + 2^k - 2^{k-i+1} - 1 < t = r + 2^k - 2^{k-i+1} + j - 1,$$

as $j \geq 1$.

Secondly, we turn to analyze the suffix distance of f. To this end, let $x, x' \in (\{0,1\}^m)^n$, $x \neq x'$, and $s = \mathsf{split}(x,x')$. We start by bounding $\mathsf{dist}\left(f(x)_{[s,n)}, f(x')_{[s,n)}\right)$ from below. If $s \in [0,r)$, set $i = 0$, $a = 0$ and $b = r - 1$. Otherwise, let $i \in [k]$, $j \in [2^{k-i}]$ be the unique pair for which $s = r + 2^k - 2^{k-i+1} + j - 1$, and set $a = r + 2^k - 2^{k-i+1}$ and $b = r + 2^k - 2^{k-i} - 1$. We have that $s \in [a,b]$ and so $x_{[a,b]} \neq x'_{[a,b]}$. Furthermore, $x_{[a,b]}$

is embedded in $f(x)_{[a,b]}$ and $x'_{[a,b]}$ is embedded in $f(x')_{[a,b]}$. If we have that $i = k$, then $s = n - 1$, and it holds trivially that

$$\mathsf{dist}\big(f(x)_{[s,n)}, f(x')_{[s,n)}\big) = 1 \geq \Delta(n-s) = \Delta.$$

So, we proceed under the assumption that $i \leq k-1$. We further set $c = r + 2^k - 2^{k-i}$ and $d = r + 2^k - 2^{k-i-1} - 1$. By the definition of z_i, we have that $x_{[a,b]} \circ z_i(x) \in \mathsf{RS}_{(m,k_i,n_i)}$ and $x'_{[a,b]} \circ z_i(x') \in \mathsf{RS}_{(m,k_i,n_i)}$, and as the two are distinct, by Fact 3.7,

$$\mathsf{dist}\big(x_{[a,b]}, x'_{[a,b]}\big) + \mathsf{dist}(z_i(x), z_i(x')) \geq n_i - k_i + 1 = 4\Delta 2^{k-i} + 1.$$

Notice that by the definition of y_i and by Claim 4.9, $\mathsf{dist}(y_i(x), y_i(x')) \geq \frac{1}{2} \cdot \mathsf{dist}(z_i(x), z_i(x'))$, and that in the construction we have that $y_i(x)$, $y_i(x')$ are embedded in $f(x)_{[c,d]}$, $f(x')_{[c,d]}$, respectively. We can thus conclude that

$$\begin{aligned}
\mathsf{dist}\Big(f(x)_{[s,n)}, f(x')_{[s,n)}\Big) &\geq \mathsf{dist}\Big(f(x)_{[a,d]}, f(x')_{[a,d]}\Big) \\
&= \mathsf{dist}\Big(f(x)_{[a,b]}, f(x')_{[a,b]}\Big) + \mathsf{dist}\Big(f(x)_{[c,d]}, f(x')_{[c,d]}\Big) \\
&\geq \mathsf{dist}\big(x_{[a,b]}, x'_{[a,b]}\big) + \mathsf{dist}\Big(f(x)_{[c,d]}, f(x')_{[c,d]}\Big) \\
&\geq \mathsf{dist}\big(x_{[a,b]}, x'_{[a,b]}\big) + \frac{1}{2}\mathsf{dist}(z_i(x), z_i(x')) \\
&\geq 2\Delta 2^{k-i} + 1,
\end{aligned}$$

where the first inequality holds trivially as, for any time t before s, $x_t = x'_t$. To conclude the proof, it only remains to bound $n - s$ from above. If it is the case that $i = 0$, then we have that $n - s \leq n$, and so by Equation (D.2), $n - s \leq 2^{k+1} = 2^{k-i+1}$. Otherwise we have that $i \geq 1$. Thus $s = r + 2^k - 2^{k-i+1} + j - 1$ and so, together with Equation (D.1), we get $n - s = 2^{k-i+1} - j \leq 2^{k-i+1}$. We conclude that

$$\begin{aligned}
\mathsf{dist}\Big(f(x)_{[s,n)}, f(x')_{[s,n)}\Big) \cdot \frac{1}{n-s} &\geq \frac{2\Delta 2^{k-i} + 1}{2^{k-i+1}} \\
&= \frac{2\Delta 2^{k-i} + 1}{2 \cdot 2^{k-i}} \\
&> \Delta,
\end{aligned}$$

as required. $\square$

Claim D.2 (Claim 4.6, restated). *Let $\Delta \in \big(0, \frac{1}{8}\big]$, $\delta \in (0,1)$, $\alpha, \beta, \gamma, k \geq 0$, and let m, ℓ*

be positive integers such that $\ell \leq 2^m$. *Then, every systematic* (α, β, γ) *tree code*

$$\mathsf{TC} : (\{0,1\}^m)^\ell \to \left(\{0,1\}^m \times \{0,1\}^k\right)^\ell$$

with distance δ *can be efficiently transformed to a systematic* $(\alpha + 8\Delta, \beta, \gamma)$ *tree code*

$$\mathsf{TC}^\Delta : (\{0,1\}^m)^\ell \to \left(\{0,1\}^m \times \{0,1\}^{k'}\right)^\ell$$

with distance (δ, Δ). *We refer to* TC^Δ *as the* Δ-suffix-distance tree code of TC.

Proof. Let

$$\mathsf{S} : (\{0,1\}^m)^\ell \to \left(\{0,1\}^m \times \{0,1\}^{8\Delta m}\right)^\ell$$

be the online function with suffix-distance Δ that is given by Claim D.1, and set $\mathsf{R}_\mathsf{S} : \{0,1\}^m \times \{0,1\}^{8\Delta m} \to \{0,1\}^{8\Delta m}$ to be the projection onto the second coordinate of S's output symbols. For a string $x \in (\{0,1\}^m)^\ell$ and $t \in [0, \ell)$, we define $\mathsf{TC}^\Delta : (\{0,1\}^m)^\ell \to \left(\{0,1\}^m \times \{0,1\}^k \times \{0,1\}^{8\Delta m}\right)^\ell$ by

$$\mathsf{TC}^\Delta(x)_t = (\mathsf{TC}(x)_t, \mathsf{R}_\mathsf{S}(\mathsf{S}(x)_t)) .$$

Observe that TC^Δ is online and systematic and that, by definition, it is an $(\alpha + 8\Delta, \beta, \gamma)$ tree code. Moreover, if TC is explicit then so is TC^Δ.

We turn to prove that TC^Δ has distance (δ, Δ). Let $x, y \in (\{0,1\}^m)^\ell$ be distinct messages with $s = \mathsf{split}(x,y)$ and let $d \in [0, n-s)$. It trivially holds that

$$\mathsf{dist}\left(\mathsf{TC}^\Delta(x)_{[s,s+d]}, \mathsf{TC}^\Delta(y)_{[s,s+d]}\right) \geq \delta(d+1).$$

Further, since TC is systematic, for every $t \in [0, \ell)$, x_t and $\mathsf{R}_\mathsf{S}(\mathsf{S}(x)_t)$ are embedded in $\mathsf{TC}^\Delta(x)_t$. Hence, the entire encoding $\mathsf{S}(x)_{[s,\ell)}$ is embedded in $\mathsf{TC}^\Delta(x)_{[s,\ell)}$. Similarly, $\mathsf{S}(y)_{[s,\ell)}$ is embedded in $\mathsf{TC}^\Delta(x)_{[s,\ell)}$. Thus

$$\mathsf{dist}\left(\mathsf{TC}^\Delta(x)_{[s,\ell)}, \mathsf{TC}^\Delta(y)_{[s,\ell)}\right) \geq \Delta\,(\ell - s) .$$

□