Explicit Binary Tree Codes with Sub-Logarithmic Size Alphabet

Inbar Ben Yaacov

Joint work with **Gil Cohen** and **Tal Yankovitz**

Tel Aviv University

November 30, 2021

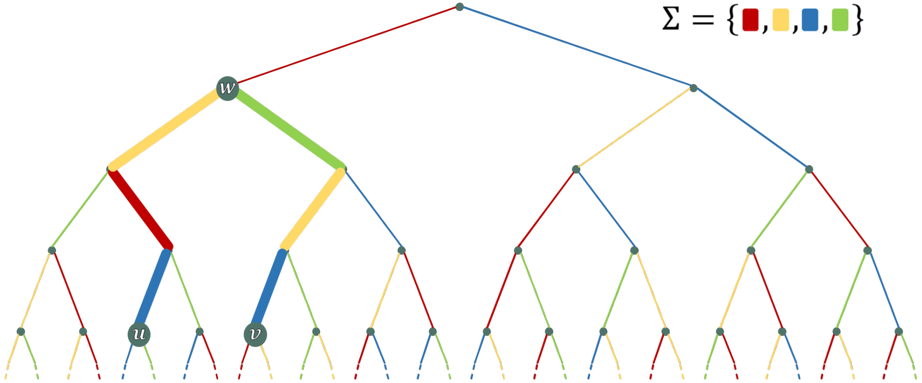
Overview

- What are tree codes?
- What are they good for?
- Known deterministic explicit constructions
- Some preparations
- Our construction
  - A closer look
- Open questions

Tree codes - definition

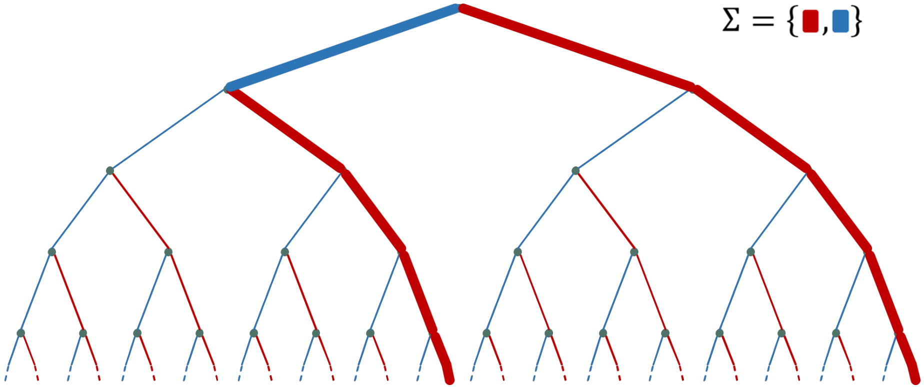
Leonard Schulman (STOC 1993)

$\Sigma = \{$🟥, 🟨, 🟦, 🟩$\}$



$$\mathsf{dist}(u, v) = \frac{2}{3}$$

Tree codes - examples

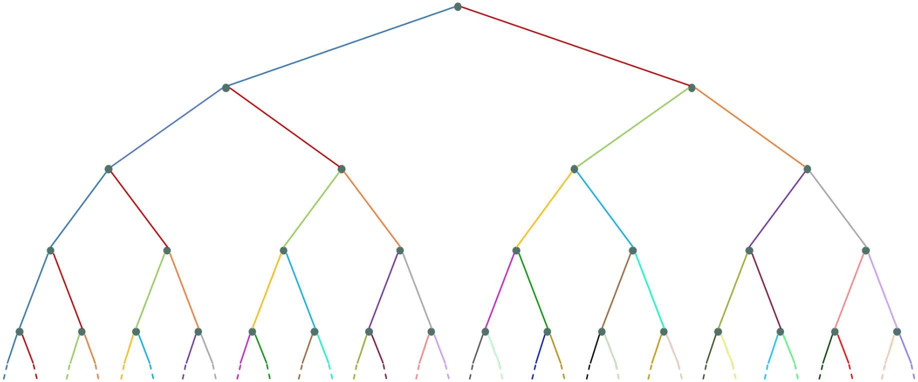


$\Sigma = \{$🟥,🟦$\}$

$$\mathsf{dist}(\mathcal{T}) = 0$$

Tree codes - examples

$\Sigma = \{0,1\}^n$, where n is the depth of the edge.



$\mathrm{dist}(\mathcal{T}) = 1$

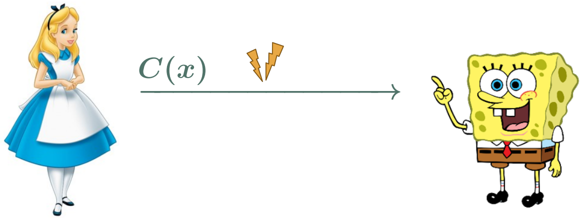
Tree codes - properties

- Number of colors, $|\Sigma|$ (smaller is better, hopefully constant).
- Distance, $\delta \in (0,1]$ (larger is better).
- Is it explicit?

Tree codes - what are they good for?

Consider the case in which Alice wishes to send Bob a message $x \in \{0,1\}^n$ over an imperfect channel.

We know that Alice can apply an ECC to her message to get $C(x) \in \{0,1\}^{\alpha n}$, for a constant α, so Bob can recover a corrupted message, even if a constant fraction of errors were occurred.

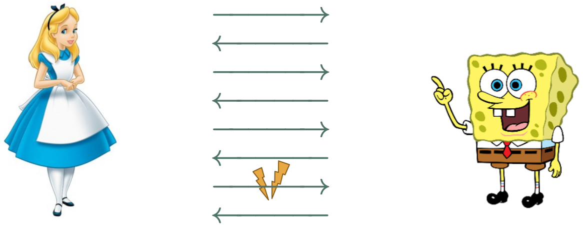


Tree codes - what are they good for?

But what happens if instead of one sending the other a single message, Alice and Bob wish to have a conversation?

In this case, the fraction of errors may occur is constant in the length of the **whole conversation**.

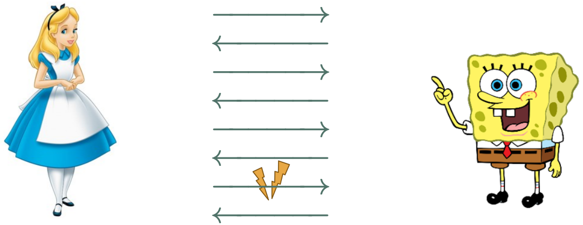
Applying a standard ECC won't suffice (why?). Example - playing chess.



Tree codes - what are they good for?

It is not at all clear that one can "pay" a constant factor in the length of the conversation, to enable Alice and Bob talk despite the errors.

Tree codes have an analog role to standard ECCs for the case of interactive communication - Schulman (STOC 1993) proved that the existence of tree codes implies resilient interactive schemes.



Asymptotically good tree codes exist!

Theorem (Schulman (STOC 1993))

For any $\delta \in (0,1)$ there exists a binary tree code with alphabet size $|\Sigma| = O_\delta(1)$ and distance δ.

Theorems (Cohen–Samocha (CCC 2020))

4 colors suffice and are necessary for constructing an asymptotically good tree code.
More specifically - there exists a 4-color tree code with distance 0.136.

Were proved by the probabilistic method...

Providing an explicit construction of an asymptotically good binary tree code is still an open problem!

Overview

- What are tree codes?
- What are they good for?
- Known deterministic explicit constructions
- Some preparations
- Our construction
    - A closer look
- Open questions

Tree codes - deterministic explicit constructions

Authors	Number of colors at depth n	Distance
Trivial	2^n	1
Evans–Klugerman–Schulman (1994)	$n^{O_\delta(1)}$	δ
Gelles–Haeupler–Kol–Ron-Zewi–Wigderson (SODA 2016)	$O(1)$	$\Omega\left(\frac{1}{\log n}\right)^*$
Cohen–Haeupler–Schulman (STOC 2018)	$(\log n)^{O_\delta(1)}$	δ
Moore–Schulman (ITCS 2014)	$O(1)$	$\Omega(1)$
BY–Cohen–Narayanan (RANDOM 2021)	$O_\delta(1)$	δ
The holy grail	$O(1)$	$\Omega(1)$

* Namely, every pair of vertices at depth n have distance $\Omega\left(\frac{1}{\log n}\right)$.

- Narayanan–Weidner (SIAM 2020) provided a construction achieving the same parameters as CHS. They also provided a randomized decoding algorithm when the correction radius is $\Omega\left(n^{-1/4}\log^{-1/2} n\right)$.

Tree codes - deterministic explicit constructions

Authors	Number of colors at depth n	Distance
Trivial	2^n	1
Evans–Klugerman–Schulman (1994)	$n^{O_\delta(1)}$	δ
Gelles–Haeupler–Kol–Ron-Zewi–Wigderson (SODA 2016)	$O(1)$	$\Omega\left(\frac{1}{\log n}\right)^*$
Cohen–Haeupler–Schulman (STOC 2018)	$(\log n)^{O_\delta(1)}$	δ
Moore–Schulman (ITCS 2014)	$O(1)$	$\Omega(1)$
BY–Cohen–Narayanan (RANDOM 2021)	$O_\delta(1)$	δ
The holy grail	$O(1)$	$\Omega(1)$

- Other works: Braverman (ITCS 2012), Pudlák (Linear Algebra Appl. 2015), Brakerski–Kalai–Saxena (FOCS 2020), Bhandari–Harsha (CoRR 2020) and many more.

Our main results

Theorem (BY–Cohen–Yankovitz (2021))

For every $\delta \in \left(0, \frac{1}{10}\right)$, there exists an explicit binary tree code with $(\log n)^{O(\sqrt{\delta})}$ colors and distance δ.

To be compared with the CHS's construction, which requires $\omega(\log n)$ colors for any constant distance δ.

Corollary

There exists an explicit binary tree code with a constant number of colors and distance $\Omega\left(\frac{1}{(\log \log n)^2}\right)$.

To be compared with the GHK$^+$'s binary tree code that has a constant number of colors and distance $\Omega\left(\frac{1}{\log n}\right)$.

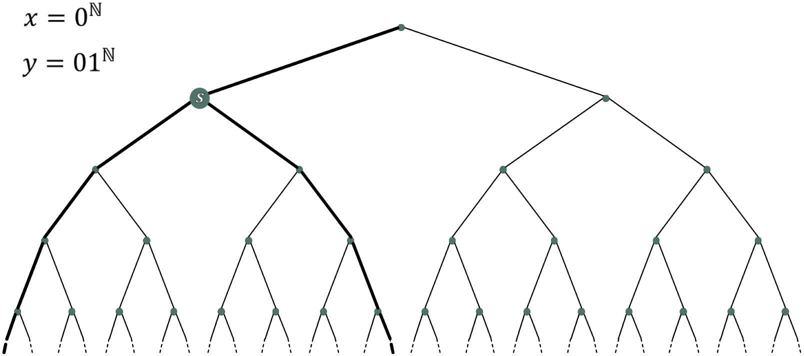
Overview

- What are tree codes?
- What are they good for?
- Known deterministic explicit constructions
- Some preparations
- Our construction
    - A closer look
- Open questions

Tree Codes - another (equivalent) definition

Definition (Split)

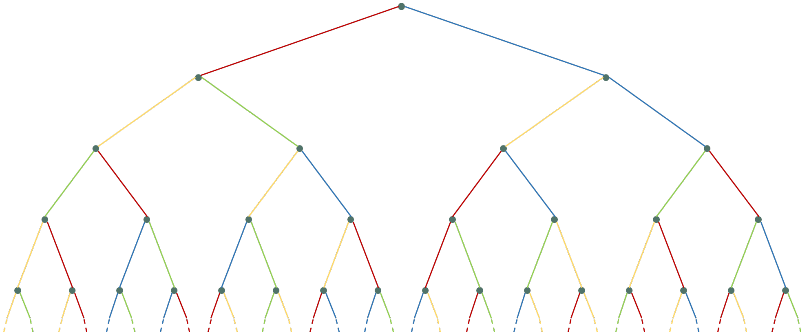
For two strings $x \neq y \in \{0,1\}^{\mathbb{N}}$ we set split(x,y) to be the minimal index i for which $x_i \neq y_i$.



Tree Codes - another (equivalent) definition

Definition (Online function)

A function $f : \{0,1\}^{\mathbb{N}} \to \Sigma^{\mathbb{N}}$ is *online* if for every $x \in \{0,1\}^{\mathbb{N}}$ and $i \in \mathbb{N}$, $f(x)_i$ is determined by $x_0, \ldots, x_i$.

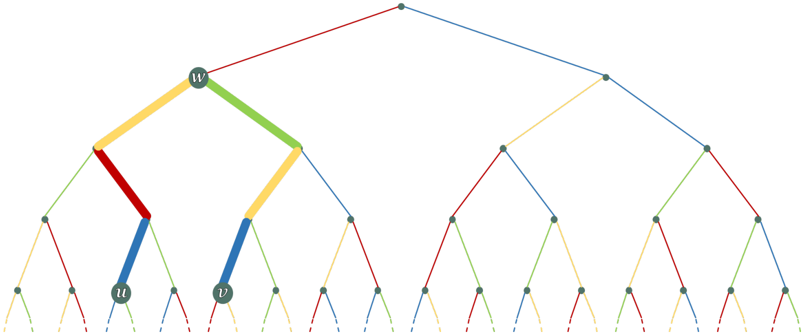


Tree Codes - another (equivalent) definition

Definition (Tree code)

An online function $\mathsf{TC} : \{0,1\}^{\mathbb{N}} \to \Sigma^{\mathbb{N}}$ is a *tree code with distance* δ if for every $x \neq y \in \{0,1\}^{\mathbb{N}}$ such that $s = \mathsf{split}(x,y)$ and every $\ell \in \mathbb{N}$,

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



Infinite Tree code $\overset{?}{\Longleftrightarrow}$ infinite family of finite tree codes

A natural relaxation is considering a tree code of **finite** depth, namely, a tree code of the form

$$\mathsf{TC} : \{0,1\}^n \to \Sigma^n,\ n \in \mathbb{N}.$$

It is clear that an infinite tree code implies finite tree codes of any finite depth - just truncate the infinite tree code at any desired depth.

Does the converse also hold? Namely, does an infinite family of finite tree codes $(\mathsf{TC}_n)_{n\in\mathbb{N}}$, such that TC_n has depth n, all have $|\Sigma|$ colors and distance δ, implies an infinite tree code (with comparable parameters)?

Apparently, the answer is **yes!**
We proved that an infinite family of finite tree codes $(\mathsf{TC}_n)_{n\in\mathbb{N}}$ with distance δ and $|\Sigma|$ colors, implies an infinite tree code with distance $\frac{\delta}{2}$ and $|\Sigma|^3$ colors.

In fact, it suffices to have a tree code of depth 2^n for every $n \in \mathbb{N}$.

The CHS construction

- Non-binary tree code (a larger arity).
- For every $m, \ell \in \mathbb{N}$, they constructed a (non-binary) tree code

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

  with distance $\frac{1}{2}$.
- By setting $m = 1$ we get a binary tree code, but then the number of colors is exponential in the depth of the tree (as in the trivial solution!).

  For comparison, the construction of Evans–Klugerman–Schulman (1994) can be used to construct a tree code with $\{0,1\}^{m \log \ell}$ colors.
- A key property: in the CHS's construction, m and ℓ have an additive relation!

CHS - reducing the alphabet binary

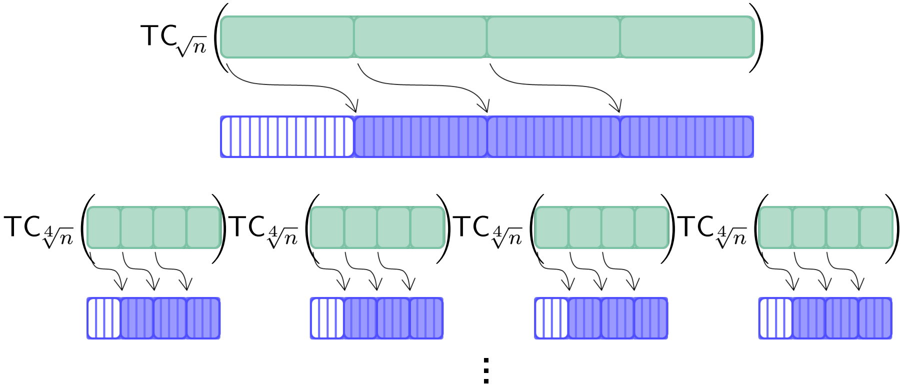
- Set $m = \ell = \sqrt{n}$ and consider

$$\mathsf{TC}_{\sqrt{n}} : \left(\{0,1\}^{\sqrt{n}}\right)^{\sqrt{n}} \to \left(\{0,1\}^{3\sqrt{n}}\right)^{\sqrt{n}}.$$

- Given a binary string $x \in \{0,1\}^n$, divide it to $\sqrt{n}$ blocks, each consists of $\sqrt{n}$ bits.

CHS - reducing the alphabet to binary, cont.

A reminder: we have $\mathsf{TC}_{\sqrt{n}} : \left(\{0,1\}^{\sqrt{n}}\right)^{\sqrt{n}} \to \left(\{0,1\}^{3\sqrt{n}}\right)^{\sqrt{n}}$

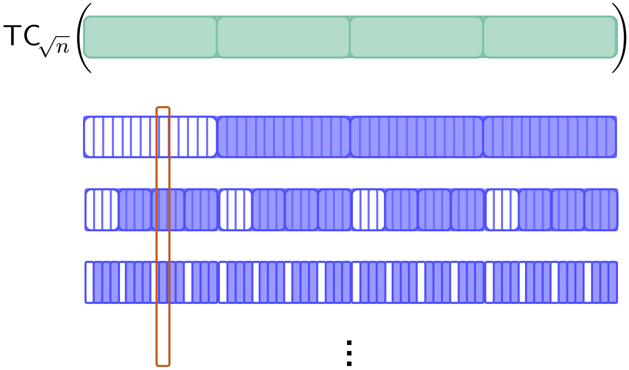


$\mathsf{TC}_{\sqrt{n}}$
$\mathsf{TC}_{\sqrt[4]{n}}$
$\mathsf{TC}_{\sqrt[4]{n}}$
$\mathsf{TC}_{\sqrt[4]{n}}$
$\mathsf{TC}_{\sqrt[4]{n}}$

...until we reach a constant block-length.

CHS - reducing the alphabet to binary, cont.

A reminder: we have $\mathsf{TC}_{\sqrt{n}} : \left(\{0,1\}^{\sqrt{n}}\right)^{\sqrt{n}} \to \left(\{0,1\}^{3\sqrt{n}}\right)^{\sqrt{n}}$



We simultaneously output the strings that were generated in all levels. Since the depth of the recursion is $\log\log n$, and for each step we write 3 bits, the alphabet size is

$$2^{3\log\log n} = (\log n)^3.$$

Overview

- What are tree codes?
- What are they good for?
- Known deterministic explicit constructions
- Some
- Our construction
  - A closer look
- Open questions

Our strategy

Strategy: reduce the dependence of the number of colors on the depth of the tree until it is entirely eliminated.

- Given a tree code

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

  with distance δ, break the number of colors to 3 components $k = (1+\alpha)m + \beta\ell^\gamma$.
  We say that TC is an $(\alpha, \beta, \gamma, \delta)$ tree code.
- Devise an efficient transformation that reduces the dependence on γ, while not deteriorating α and β too much, and further, not harming the distance δ **(at all!)**.
- Apply the transformation iteratively.

Our transformation

Theorem (The γ reduction transformation)

There exists an efficient transformation that given an $(\alpha, \beta, \gamma, \delta)$ tree code

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

it transforms it to an $\left(\alpha + \sqrt{\delta}, \beta + \sqrt{\delta}, \frac{\gamma}{2}, \delta\right)$ tree code

$$\mathsf{TC}_{\mathsf{out}} : (\{0,1\}^m)^{\ell^2} \to \left(\{0,1\}^{(1+\alpha+\sqrt{\delta})m+(\beta+\sqrt{\delta})\ell^{\frac{\gamma}{2}}}\right)^{\ell^2}.$$

* The transformation requires $m \geq \log \ell$. Thus, we take $m = \log \ell$, and to reduce the alphabet to binary, we use a well-known and efficient transformation that has almost no cost in parameters and requires $m = O(\log \ell)$.

Applying the transformation iteratively

Reminders: Breaking the number of colors to components: $k = (1+\alpha)m + \beta\ell^{\gamma}$,
The transformation: $(\alpha, \beta, \gamma, \delta) \longmapsto \left(\alpha + \sqrt{\delta}, \beta + \sqrt{\delta}, \frac{\gamma}{2}, \delta\right)$.

- In our notations,

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

  is a $\left(1, 1, 1, \frac{1}{2}\right)$ tree code.

- By a simple observation, we deduced a variation, $\mathsf{TC}^{\delta}_{m,\ell}$, that is a $(\delta, \delta, 1, \delta)$ tree code.
- Set $\mathsf{TC}_{\mathsf{in}} = \mathsf{TC}^{\delta}_{m,\ell}$. Applying the transformation yields a $\left(\delta + \sqrt{\delta}, \delta + \sqrt{\delta}, \frac{1}{2}, \delta\right)$ tree code.
- Feeding the resulted tree code to the transformation yields a $\left(\delta + 2\sqrt{\delta}, \delta + 2\sqrt{\delta}, \frac{1}{4}, \delta\right)$ tree code.

Applying the transformation iteratively, cont.

Reminders: Breaking the number of colors to components: $k = (1+\alpha)m + \beta\ell^{\gamma}$,
The transformation: $(\alpha, \beta, \gamma, \delta) \longmapsto \left(\alpha + \sqrt{\delta}, \beta + \sqrt{\delta}, \frac{\gamma}{2}, \delta\right)$.

- Doing this process iteratively for r times, yield an $\left(r\sqrt{\delta}, r\sqrt{\delta}, \frac{1}{2^r}, \delta\right)$ tree code.
- By setting $r \approx \log\log\ell$, we eliminated the dependence on the depth and obtained a distance-δ tree code

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

-

$$\Longrightarrow \ \mathsf{TC}_{\mathsf{bin}} : \{0,1\}^\ell \to \left(\{0,1\}^{O(\sqrt{\delta}\log\log\ell)}\right)^\ell$$

Our binary tree codes

Summary:

- $\mathsf{TC}^{\delta}_{m,\ell}$ is explicit, our transformation is efficient, we apply it $\log\log\ell$ times, the reduction to binary alphabet is efficient, and $\implies$ $\mathsf{TC}_{\mathsf{bin}}$ is explicit.
- $\mathsf{TC}^{\delta}_{m,\ell}$ has (a constant) distance $\delta > 0$ and our transformation preserves it $\implies$ $\mathsf{TC}_{\mathsf{bin}}$ has distance $\Omega(\delta) > 0$.
- $\mathsf{TC}_{\mathsf{bin}}$ has
$$2^{O(\sqrt{\delta}\log\log\ell)} = (\log\ell)^{O(\sqrt{\delta})}$$
colors.

Overview

- What are tree codes?
- What are they good for?
- Known deterministic explicit constructions
- Some preparations
- Our construction
    - A closer look
- Open questions

Diving into our transformation

A reminder: we want to transform a distance-δ tree code

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

to

$$\mathsf{TC}_{\mathsf{out}} : (\{0,1\}^m)^{\ell^2} \to \left(\{0,1\}^{(1+\alpha+\sqrt{\delta})m+(\beta+\sqrt{\delta})\ell^{\frac{\gamma}{2}}}\right)^{\ell^2}.$$

without deteriorating the distance.

Diving into the transformation, cont.

- Given a message $x = (x_1, x_2, \dots, x_{\ell^2}) \in \left(\{0,1\}^m\right)^{\ell^2}$, write it in an $\ell \times \ell$ matrix.
- Apply $\mathsf{TC}_{\mathsf{in}} : \left(\{0,1\}^m\right)^\ell \to \left(\{0,1\}^{(1+\alpha)m+\beta\ell^\gamma}\right)^\ell$ to each rows.
- Apply the variation on the CHS construction

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

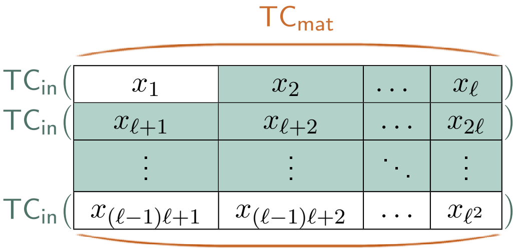
to the **entire matrix**, and apply an ECC with distance $\sqrt{\delta}$ on each of its output symbols. (Why not tensoring?)

$$\begin{array}{r|c|c|c|c|l}
 & \multicolumn{4}{c}{\overbrace{\hspace{16em}}^{\mathsf{TC}_{\mathsf{mat}}}} & \\
\hline
\mathsf{TC}_{\mathsf{in}}\big(& x_1 & x_2 & \dots & x_\ell & \big) \\
\hline
\mathsf{TC}_{\mathsf{in}}\big(& x_{\ell+1} & x_{\ell+2} & \dots & x_{2\ell} & \big) \\
\hline
 & \vdots & \vdots & \ddots & \vdots & \\
\hline
\mathsf{TC}_{\mathsf{in}}\big(& x_{(\ell-1)\ell+1} & x_{(\ell-1)\ell+2} & \dots & x_{\ell^2} & \big) \\
\hline
 & \multicolumn{4}{c}{\underbrace{\hspace{16em}}} &
\end{array}$$

Distance analysis

Distance analysis:

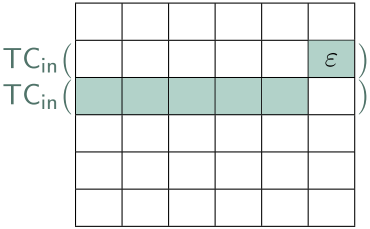
- If the entire "test" is contained in a single row, $\mathsf{TC}_{\mathsf{in}}$ guarantees distance δ.
- If the entire "test" contains full rows, $\mathsf{TC}_{\mathsf{mat}}$ guarantees distance $\sqrt{\delta}$ (and together with the ECC, distance δ).
- What if we have neither?



An example

In this example, $x = \dots 00\varepsilon 00 \dots$ and ε is located at the end of the row.

- $\mathsf{TC}_{\mathsf{in}}\left(x^{(2)}\right)$ gives a vanishing distance of $\frac{1}{\ell}$, and $\mathsf{TC}_{\mathsf{in}}\left(x^{(3)}\right)$ guarantees no distance.
- Further, since the test doesn't include a full row, we don't have any fully written output symbol of $\mathsf{TC}_{\mathsf{mat}}$ after the split.
- We need a solution that provides more distance as the split gets closer to the end of the row.



Distance analysis, cont.

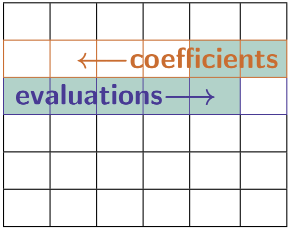
A partial solution: a "reverse" polynomial.

For each row $i \in [\ell]$, we define a polynomial that read its coefficients from the previous row. More formally:

$$g_i(T) = \sum_{j=1}^{\ell} x_{i-1,j} T^{\ell-j} \in \mathbb{F}_{2^m}[T].$$

Note that it reads the coefficients **"backwards"** - from right to left.

Then, we evaluate g_i over distinct (but fixed) field elements, and write the evaluations in the respective row.

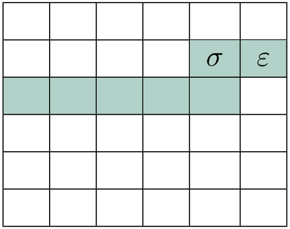


An example

In this example,

$$g_3(T) = \sigma T + \varepsilon.$$

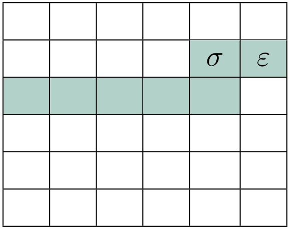
Note that if there are not many cells in the upper row, g_3 is a small degree polynomial (with degree $= s - 1$). Hence, it has few roots in the consecutive row to the split.



Another example

But what is all the roots of g_i are happened to be at the beginning of the consecutive row, and the test doesn't include enough cells?

In this case, we must rely on $\mathsf{TC}_{\mathsf{in}}$, but then the distance will deteriorate...



Distance analysis, cont.

The solution: suffix distance.

We equip TC_{in} with another distance property that provides it a larger distance only if we read the end of the message. More formally:

Definition (Suffix distance)

A tree code $\mathsf{TC} : \Sigma^\ell \to \Pi^\ell$ has suffix distance Δ if for every $x \neq y \in \Sigma^\ell$ with $s = \mathsf{split}(x,y)$, $\mathsf{dist}\Big(\mathsf{TC}(x)_{[s,\ell]}, \mathsf{TC}(y)_{[s,\ell]}\Big) \geq \Delta$.

We (efficiently) transform TC_{in} to an $(\alpha + \Delta, \beta, \gamma, \delta)$ tree code with suffix distance Δ, $\mathsf{TC}^{\Delta}_{\text{in}}$.

$$\begin{array}{r}
\mathsf{TC}^{\Delta}_{\text{in}} \\
\mathsf{TC}^{\Delta}_{\text{in}} \\
\\
\mathsf{TC}^{\Delta}_{\text{in}}
\end{array}
\overbrace{\underbrace{\begin{array}{|c|c|c|c|}
\hline
x_1 & x_2 & \dots & x_\ell \\ \hline
x_{\ell+1} & x_{\ell+2} & \dots & x_{2\ell} \\ \hline
\vdots & \vdots & \ddots & \vdots \\ \hline
x_{(\ell-1)\ell+1} & x_{(\ell-1)\ell+2} & \dots & x_{\ell^2} \\ \hline
\end{array}}_{}}^{\mathsf{TC}^{\sqrt{\delta}}_{m\ell,\ell}}$$

Distance analysis, cont.

Thus, to have distance δ, at time $(i, j) \in [\ell] \times [\ell]$ we output

- $\mathsf{TC}^{\Delta}_{\mathsf{in}}\left(x^{(i)}\right)_j$, that handles single row tests and two-row tests, where the split is at the beginning of the row.
- $\mathsf{TC}_{\mathsf{mat}}(x)_{i-1}$, that handles multiple rows tests.
- g_i, that handles two-row tests, where the split is at the end of the row.

The resulted distance is the **minimum** over the distance each of above guarantees (recall that we apply an ECC with to the output symbols of $\mathsf{TC}_{\mathsf{mat}}$).

x_1	x_2	$\dots$	x_ℓ
$x_{\ell+1}$	$x_{\ell+2}$	$\dots$	$x_{2\ell}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$
$x_{(\ell-1)\ell+1}$	$x_{(\ell-1)\ell+2}$	$\dots$	x_{ℓ^2}

Analyzing the number of colors

A problem: What happens if we output both the outputs of $\mathsf{TC}_{\mathsf{in}}^{\triangle}$ and $\mathsf{TC}_{\mathsf{mat}}$?

- Recall that each tree code outputs at least one bit per every bit read.
- Thus, In the least worst case, applying the transformation to the CHS's $(\delta, \delta, 1, \delta)$ tree code, would yield an
$$\left(\boldsymbol{\delta + 1}, \delta + \sqrt{\delta}, \frac{1}{2}, \delta\right)$$
tree code.
- After r iterations, we will get an $\left(\boldsymbol{\delta + r}, \delta + r\sqrt{\delta}, \frac{1}{2^r}, \delta\right)$ tree code. Thus, the resulted binary tree code will be of the form
$$\mathsf{TC}_{\mathsf{bin}} : \{0,1\}^{\ell} \to \left(\{0,1\}^{O(\log\log \ell)}\right)^{\ell},$$
resulting in at least $\omega(\log \ell)$ colors.

analyzing the number of colors, cont.

Remarks:

- This is also the reason we couldn't afford using the $\left(1, 1, 1, \frac{1}{2}\right)$ CHS construction, but had to relax it to a $(\delta, \delta, 1, \delta)$ tree code.
- This restriction also prevented us from using a standard "shifting" solution for dealing with the case of a test that includes 2 non-full rows.

Analyzing the number of colors, cont.

Solution: systematic encodings.

Informally, we say that an encoding is *systematic* if its input appears in the output. For example, $f(x) = (x, y)$.

Since the CHS construction is systematic, and both $\mathsf{TC}_{\mathsf{in}}^{\Delta}$ and $\mathsf{TC}_{\mathsf{mat}}$ are applied to the original input, we can output the systematic part only **once**, and output only the **redundant** parts of $\mathsf{TC}_{\mathsf{in}}^{\Delta}$, $\mathsf{TC}_{\mathsf{mat}}$ and the ECC applied to it, while making sure that all of the above are not too "pricey".

Analyzing the number of colors, cont.

A problem: What happens if we output an evaluation of g_i for each cell?

Since we need ℓ distinct elements, we work over a filed of size $\geq \ell = 2^m$. Thus we need at least m bits for writing an evaluation. This results in the exact same problem as in the previous case.

Solution: we output a symbol only once per c symbols read, and "spread" it over c output symbols. Here again we need to use a block ECC, hence we set $c \approx \frac{1}{\sqrt{\delta}}$ and used an ECC with distance $\sqrt{\delta}$.

Conclusion

With these solutions in hand, the number of colors required for the resulted tree code is

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

and by applying the transformation iteratively, where the initial input is the $(\delta, \delta, 1, \delta)$ CHS's construction, we get the desired result.

Open questions

- The holy grail - devise a construction with a constant distance and a constant number of colors.
- Breaking the $\log \ell$ barrier for a high constant distance, say $\delta = 0.9$ with $\sqrt{\log \ell}$ colors.
- Is there a better transformation that can reduce the number of colors used by CHS even further?
- Devise a construction with constant distance and $\mathrm{poly}(\log \log \ell)$ colors.

Thank you for listening!