CS 229r Essential Coding Theory Apr 27, 2020

Lecture 24

Instructor: Madhu Sudan *Scribe: Yash Nair*

1 Outline: Coding for Interaction

- Set up interaction and coding architecture
- History
- Tree Codes
- Braverman-Rao Protocol

2 Interaction

We first define a framework for interaction. Consider the following dialogue:

- Alice says to Bob, "Please explain Fermat's Theorem"
- Bob explains group theory, Galois theory, modular forms, and semi-stable representations to Alice
- Alice tells Bob that she meant Fermat's *Little* Theorem

The above illustrates how information can be wasted if an error occurs. Bob wasted a lot of time communicating irrelevant information to Alice: in this interactive setting, a few undetected errors can completely derail communication and interaction.

We now define the notion of an interaction.

Definition 1 (Interaction)**.** *An interaction is a subgraph of a rooted binary tree. Alice's input is, for every node at even depth, one edge pointing to one of the child nodes. Bob's input is, for every node at odd depth, one edge pointing to one of the child nodes. The 'message' corresponding to an interaction is the path of nodes traversed by following the edges of Alice and Bob's interaction.*

We now formally define a binary protocol of length k as follows.

Definition 2 (Binary Protocol)**.** *Let $V = \{0,1\}^{\leq k}$, and call V the stage. Partition $V = V_A \dot\cup V_B$ into the set of vertices from which Alice communicates, V_A, and the set of vertices from which Bob communicates, V_B. Define*

$$E = \{(x, x0), (x, x1) | x \in \{0,1\}^{<k}\},$$

and partition $E = E_A \dot\cup E_B$, where E_A is the set of edges with head in V_A, and E_B is the set of edges with head in V_B. An instance of the communication problem is given by subsets $\Pi_A \subset E_A, \Pi_B \subset E_B$ such for any vertex, v, in V_A (V_B) there is exactly one edge in Π_A (Π_B) with v as its head. The objective of the communication problem is to output a valid transcript (i.e. path) from the root $\emptyset$ to a leaf $m \in \{0,1\}^k$ such that m is A-valid and m is B-valid (where we say that m is A-valid (B-valid) if the path is a subset of $\Pi_A \cup E_B$ ($\Pi_B \cup E_A$)).

It is clear that, given Π_A and Π_B, there is a unique transcript, and that this transcript can be communicated, between Alice in Bob, using only k bits in the noiseless setting.

3 Interactive Coding

Now that we have defined a framework for interaction, we define the notion of interactive coding.

Definition 3 (Interactive Code)**.** *An interactive code is given by two encoding functions*

$$E_A : \Pi_A \to \Gamma_A \text{ and } E_B : \Pi_B \to \Gamma_B,$$

where (Γ_A, Γ_B) *are edges on* $\{0,1\}^{\leq n}$ *(with the property that for any vertex* $v \in \{0,1\}^{\leq n}$ *at an even (odd) depth, there is exactly one edge in* Γ_A *(*Γ_B*) with* v *as its head), and two functions*

$$f_A, f_B : \{0,1\}^n \to \{0,1\}^k.$$

We say that the coding scheme can correct e *errors if an adversary gives Alice and Bob paths* a *and* b *in* $\{0,1\}^n$*, respectively, such that* a *is* A*-valid,* b *is* B*-valid,* $\Delta(a,b) \leq e$*, and*

$$f_A(a) = f_B(b) = m,$$

the original interaction.

In 1990, Schulman introduced the problem, a concept called "Tree Codes", and showed an $\Omega(1)$-rate protocol correcting $\Omega(1)$-fraction errors. Braverman and Rao, in 2011 then showed an $\Omega(1)$-rate protocol correcting $\frac{1}{8}$-fraction errors, which is nearly optimal.

3.1 Tree Codes

We now define a tree code, which is a coding scheme describing how to encode every binary string.

Definition 4. *A tree code,* T*, is an online code with,*

$$T : \{0,1\}^* \to \{0,1\}^*$$

to be such that $|x| = |y| \implies |T(x)| = |T(y)|$ *and such that,* $\forall x \in \{0,1\}^*$*,* $T(x)$ *is the prefix of* $T(x0)$ *and* $T(x1)$*. We define the rate of such a code to be* R *such that* $|T(x)| \leq \frac{k}{R}$ *for all* $x \in \{0,1\}^k$*. In fact, here we will only consider* $|T(x)| = c|x|$ *so that* $R = \frac{1}{c}$*, with* $c \in \mathbb{N}$*. We say that the distance of such a code is* δ *if*

$$\forall x, y \in \{0,1\}^k, i \in [k], x_i \neq y_i \implies \Delta(T(x), T(y)) \geq \frac{\delta(k-i+1)}{R}.$$

Schulman showed, using tree codes, that it is possible to achieve $\delta, R = \Omega(1)$, and that this is useful for interactive coding. We focus on the former result.

Theorem 5 (Schulman 1990)**.** *There exist tree codes with* $\delta, R = \Omega(1)$*.*

Proof. We will take T to be a linear code with an almost upper-triangular generator. Specifically, T will be of the form

$$\begin{bmatrix} * & * & * & * & * & * & * & * & * & \cdots \\ 0 & 0 & 0 & * & * & * & * & * & * & \cdots \\ 0 & 0 & 0 & 0 & 0 & 0 & * & * & * & \cdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix},$$

where we choose the $***$'s at random in such a way that matrix is Toeplitz. The rest of the proof is left as an exercise. □

Exercise 6. *Recursively construct the above generator matrix as follows. Define*

$$G_{2k} = \begin{bmatrix} G_k & H_k \\ 0 & G_k \end{bmatrix},$$

where

$$G_k \in \mathbb{F}_2^{k \times ck},$$

for some constant $c \in \mathbb{N}$, and

$$H_k \in \mathbb{F}_2^{k \times ck},$$

is a random matrix. Then

$$P(G_{2k} \text{ doesn't work} | G_k \text{ works}) = \exp(-k).$$

The above exercise motivates the next:

Exercise 7. *Over large alphabets, codes with $\delta \to 1$ exist with $R > 0$*

At a high level, Schulman's protocol involves local moves on Π_A and Π_B. After each round of communication, Alice and Bob check if they agree to ensure that no errors have occurred; however, if an error occurs, they backtrack to an agreed upon point. In contrast to this, the Braverman-Rao protocol makes global moves–instead of backtracking, Alice and Bob continue communicating without backtracking, but in a way that has changed due to their knowledge of potential error.

3.2 Braverman-Rao Protocol

We now describe the Braverman-Rao Protocol. Throughout, i will denote the round index for Alice, and j, the round index for Bob.

- Fix tree codes T^A and T^B. They need not be the same, but both Alice and Bob have access to the other's tree code.
- We now describe Alice's protocol:
  - Maintain a sequence of edges $(e_1 ... e_i) = S_A \subset \Pi_A$ of her own edges. These are the edges that Alice has told Bob about. Then at each round, Alice compresses this sequence (we describe this later), encodes with T^A, and sends the new bits to Bob
  - At round j:
    - Alice receives m_j^B from Bob, and combines with past messages to get $m_{\leq j}^B$
    - Alice brute force decodes $m_{\leq j}^B$ to get $R_{B,j}$ (where, here, by decoding, we mean that Alice finds the $R_{B,j}$ that minimizes $\Delta(m_{\leq j}^B, T^B(R_{B,j}))$)
    - If $R_{B,j}$ seems like Bob is suggesting to take irrelevant edges based on his previous choices, or he suggests to take both edges out of the same node, ignore the transmission and move on to the next stage
    - Otherwise, if the next edge e that Alice wants to take after following the unique path in $S_A \cup R_{B,j}$ is not in S_A, then add $e_{i+1} \leftarrow e$ to S_A
  - Send $(i+1)$th bit of $T^A(S_A)$ to Bob
  - Repeat the above for n steps
  - If the above determines a path in (Π_A, Π_B), return it.

In the above, the question remains of how we can efficiently compress writing an edge without needing k bits (to specify the child node, for example). Notice that we add edge e_i only if Alice had previously added e_j to S_A. Thus, we can encode e_i as (j, b_1, b_2) where b_1 indicates the edge that Bob took from the tail of e_j, and b_2 indicates that edge that Alice takes from there. Since $|S_A| = O(k)$ it takes $O(\log k)$ bits to describe

j, and thus using this compression it takes only $O(\log k)$ bits to write transmit an edge. To get even better compression, we only need send $(i-j, b_1, b_2)$, since both players know the round number, i. The advantage here is that $i-j$ is typically constant (in expectation). We now give a high-level overview of the analysis of this protocol.

3.3 Analysis of Braverman-Rao Protocol

At a high level, we will measure progress by S_A and S_B, asking how far down the correct path (given by $\Pi_A \cup \Pi_B$) $S_A \cup S_B$ gets. The strength of the Braverman-Rao construction, in that it does not erase progress is, that progress can never decrease. In particular, we may ask, when the path length given by $S_A \cup S_B$ is i, and Alice has just added the ith edge to S_A, why Bob does not add the $i+1$th edge. The only thing preventing Bob from adding the $i+1$th edge is the errors of the adversary. But since the adversary is limited in its errors, if Alice and Bob keep communicating, Bob will eventually add the $i+1$th edge to S_B. There will be a constant number of steps between path length being i and path length being $i+1$ due to the fact that the $i+1$th edge must be communicated and the number of errors that the adversary has introduced.

With this high-level overview, we give a summary of the analysis in Braverman-Rao. First, they introduce three key definitions:

Definition 8. *Define $N(i,j)$ to be the number of errors between the ith and jth rounds of communication. Define $m(i)$ to be the length of the prefix in $\Gamma_A \cup \Gamma_B$ agreed upon by Alice and Bob after i transmissions in $\Gamma_A \cup \Gamma_B$. Finally, define $t(i)$ to be the first time that the ith edge enters S_A.*

With these definitions, Braverman and Rao prove three claims; we give some intuition for each:

- $m(n) \geq t(k)$. This is due to the fact that an error case (i.e. $f_A(a) \neq f_B(b)$) only occurs when $m(n) < t(k)$.
- $m(t(i)-2) < t(i-1)$. The intuition for this statement is that a disagreement on the ith edge is due to a prior inconsistency which occurred before the $(i-1)$th edge was announced.
- $N(m(i)+1, i) \geq \delta(i - m(i))/2$. Intuitively, this claim states that the inconsistency is contributed by the error.

These three claims together imply that the Braverman-Rao protocol corrects $\delta/2$ fraction errors.

3.4 Constructability

Notice that none of the above protocols are constructive. In fact, the constructability of tree codes would imply that of the Schulman protocol which would imply that of the Braverman-Rao protocol. Thus, we do not yet have deterministic efficient protocols nor an explicit construction of a tree code.