CS 229r Essential Coding Theory April 20, 2020

Lecture 22

Instructor: Madhu Sudan *Scribe: Joyce Tian*

1 Pseudorandomness

First, we will introduce the notion of pseudorandomness. A pseudorandom generator (PRG) G deterministically maps from $\{0,1\}^S \to \{0,1\}^n$ such that $S \leq n$. Specifically, we will use PRGs to minimize the amount of pure random bits (length S) needed to simulate creation of longer strings (length n) such that the PRGs ε-fool algorithms.

Definition 1 ("$\varepsilon - fooling$")**.** Given some algorithm A which expects inputs x and r, a random string of length n, the PRG G ε-fools A, where $A(x;r) \in \{0,1\}$, if for uniformly randomly distributed binary strings R and z,

$$|\Pr_r[A(x,r)=1] - Pr_z[A(x,G(z))=1]| \leq \varepsilon.$$

Calling on prior lecture definitions of ε-approximate, this means

$$\Pr_R[A(x,r)=1] \approx_\varepsilon \Pr_z[A(x,G(z))=1].$$

Note that given some randomized algorithm A which takes as input n random bits and has runtime T and a PRG $G : \{0,1\}^S \to \{0,1\}^n$ which ε-fools A, we can produce deterministic algorithm with $2^S \cdot T$ runtime (as there are 2^S possible inputs for G).

While we would ideally like to prove that such PRGs exist for all polytime algorithms, we will focus first on some examples.

2 Max t-SAT

2.1 Randomized Algorithm

We will start by introducing the max t-SAT problem. Consider an input of m clauses $C_1, \ldots, C_m$ which use n total variables such that each clause has t distinct variables, $C_j = x_{j_1} \vee x_{j_2} \vee \cdots \vee x_{j_t}$. If so, max t-SAT outputs assignments $\ell_1, \ldots, \ell_n$ satisfying as many clauses as possible.

For a benchmark, we consider the Bernoulli random assignment. This is expected to yield $(1-2^{-t})\,m$ satisfied clauses, since within each clause each variable has i.i.d $1/2$ chance of being correctly filled. (Note that because we find this by linearity of expectation, there is no need for independence between clauses, and should we so desire we could repeat an input clause.)

Exercise 2. *Find a randomized algorithm which outputs* $(1-2^{-t})\,m$ *satisfied clauses with high (exponentially-small) probability.*

2.2 Deterministic Algorithm with t-wise Independent PRG

Now, we would like to find a deterministic PRG which approximates the optimal solution of max t-SAT. For this, we will define a few terms:

Definition 3 (α-approximator)**.** An algorithm A' α-approximates A if for all inputs x, $A'(x)$

Definition 4 (t-wise independence)**.** A deterministic function $G : \{0,1\}^S \to \{0,1\}^n$ is t-wise independent if for all $T \subseteq [n]$ such that $|T| \leq t$,

$$G(z)|_T \sim \text{Unif}(\{0,1\}^T)$$

If there exists a t-wise independent $G : \{0,1\}^S \to \{0,1\}^n$, then we can use this G to yield a deterministic $(1-2^{-t})\,m$ approximator to max t-SAT which has runtime around $2^S m$. Specifically, we consider all 2^S possible inputs to G and take around m time to check the clause fulfillments. Now, we want to find such a t-wise independent G such that $|S|$ is as small as possible and n is as large as possible.

Lemma 5. For an $[n,k,d]_2$-linear code C such that $\Delta(C^\perp) > t$, the encoder of C, $E : \mathbb{F}_2^k \to \mathbb{F}_2^n$ is t-wise independent.

Proof. Let $S \subseteq [n]$ denote the column numbers in E^T, the parity-check matrix of $C^\perp$, which form the smallest set of linearly dependent columns, which must exist since the dual code does not have infinite distance. Then, we construct vector $y \in \mathbb{F}_2^n$ which is 0 in all coordinates except those in S, at which it is 1. Necessarily $E^T y = 0$ and thus $y \in C^\perp$. Since $C^\perp$ has a distance of $\Delta(C^\perp) > t$ and $0^n \in C^\perp$ by definition, we note that necessarily this means that $|S| \geq t+1$. If so, then this means that any t columns in G^T are independent, and thus any t rows in E are independent. □

Thus, we have reframed to searching for the smallest linear code C such that its dual $C^\perp$ has distance at least t, as this would allow for us to generate the best runtime per our findings above.

We note that the largest simple linear code whose distance is t which we have encountered so far is the BCH code of distance $t+1$, which would be a $[n, n-\frac{t}{2}\log(n), t+1]_2$-code. Letting this be $C^\perp$, we thus have that C is a code with encoder $E : \mathbb{F}_2^{\frac{t}{2}\log(n)} \to \mathbb{F}_2^n$, which we will thus use as a PRG. We note that this PRG's image is $2^{\frac{t}{2}\log(n)} = n^{t/2}$, meaning that we have found a deterministic $n^{\frac{t}{2}}$-approximator for max t-SAT.

2.3 Deterministic Algorithm with δ-almost t-wise Independent PRG

Having found a solution which is exponential in t and polynomial in n, we would now like to find something which is polynomial in t as well.

Definition 6 (ε-bias)**.** A PRG $G : \mathbb{F}_2^S \to \mathbb{F}_2^n$ is ε-biased if for all linear functions $L : \mathbb{F}_2^n \to \mathbb{F}_2$,

$$\left|\Pr_{r\sim \mathrm{Unif}(\{0,1\}^n)}[L(r)=1] - \Pr_{z\sim \mathrm{Unif}(\{0,1\}^S)}[L(G(z))=1]\right| \leq \varepsilon$$

Definition 7 (δ-almost t-wise independence)**.** A PRG $G : \mathbb{F}_2^S \to \mathbb{F}_2^n$ is δ-almost t-wise independent if for all $T \subseteq [n]$ such that $|T| \leq t$, the total variation distance between $\{G(z)|_T\}_{z\sim \mathrm{Unif}(\{0,1\}^S)}$ and $\mathrm{Unif}(\{0,1\}^T)$ is bounded by δ, which we can denote as

$$\{G(z)|_T\}_{z\sim \mathrm{Unif}(\{0,1\}^S)} \approx_\delta \mathrm{Unif}(\{0,1\}^T)$$

Exercise 8. *Prove that if we have a δ-almost t-wise independent PRG $G : \mathbb{F}_2^S \to \mathbb{F}_2^n$, then we have a $(1-2^{-t}(1-\delta))$-approximator to max t-SAT with runtime $2^S \cdot m$.*

We will then consider Vazirani's XOR lemma, which says the following:

Lemma 9. If $G : \mathbb{F}_2^S \to \mathbb{F}_2^n$ is ε-biased, then

$$G(z) \approx_{2^n\cdot\varepsilon} \mathrm{Unif}(\{0,1\}^n)$$

Proof. WLOG any linear function $L : \mathbb{F}_2^n \to \mathbb{F}_2$ can be written as $L(x_1,\ldots,x_n) = \sum_{i=1}^n \alpha_i x_i$ where $\alpha_i \in \mathbb{F}_2$. Thus, for any nonzero linear function L, $\Pr_{x\sim \mathrm{Unif}}[L(x)=1] = \frac{1}{2}$ since if so $L(x) \sim \mathrm{Unif}(0,1)$. Then, per the definition of ε-bias,

$$\left|\Pr_{z\sim \mathrm{Unif}}[L(G(z))=1] - \frac{1}{2}\right| \leq \varepsilon.$$

Specifically, we note from this that $L(G(z))$ is 0 with probability $\leq \frac{1}{2}+\varepsilon$, and similarly for 1. From this, we find that for nonzero L,

$$\left|E_{z\sim \mathrm{Unif}}\left[(-1)^{L(G(z))}\right]\right| = |\Pr_z[L(G(z))=0] - Pr_z[L(G(z))=1]| \leq 2\varepsilon,$$

since there can only be up to 2ε difference in the probability of 0 and 1.

From these findings, we aim to prove that for all $y \in \mathbb{F}_2^n$,

$$\left|\Pr[G(z) = y] - 2^{-n}\right| \leq 2 \cdot \varepsilon.$$

It seems initially unintuitive to derive a bound on probabilities for specific outputs of the PRG when our only bound so far has been on the PRG's image. To show this, we first define the indicator function

$$\delta_y(x) = \frac{1}{2^n} \sum_L (-1)^{L(x-y)} = E_L[(-1)^{L(x-y)}],$$

which we can inspect to find yields 1 if $x = y$ and 0 otherwise.

Now, we note from above that for nonzero L,

$$\left|E_{z \sim \text{Unif}}\left[(-1)^{L(G(z))}\right]\right| \leq 2\varepsilon.$$

From this, we find that

$$\left|E_L\left[E_{z \sim \text{Unif}}\left[(-1)^{L(G(z))}\right]\right] - 2^{-n}\right| \leq 2\varepsilon,$$

since $L = 0$ with probability 2^{-n} and if so would yield an inner expectation of 1. Exchanging the expectations (since we note these must be finite values), we see that the new inner expectation is simply an indicator function

$$\left|E_{z \sim \text{Unif}}\left[\delta_y(G(z))\right] - 2^{-n}\right| \leq 2\varepsilon.$$

This is equivalent to claiming that for all $y \in \mathbb{F}_2^n$,

$$\left|\Pr[G(z) = y] - 2^{-n}\right| \leq 2 \cdot \varepsilon,$$

and thus we have found the aforementioned bound. Now, we note that there are 2^n possible y-values, and thus by the union bound,

$$\sum_y \left|\Pr[G(z) = y] - 2^{-n}\right| \leq 2 \cdot \varepsilon \cdot 2^n.$$

□

Lemma 10. If G is ε-biased, then for all $t \leq n$, G is $(\varepsilon \cdot 2^t)$-almost t-wise independent.

Proof. Since G is ε-biased, we note that for any $t \leq n$, for all $T \subseteq [n]$ such that $|T| \leq t$, $G|_T$ is ε-biased as well. Then, per Lemma 9, we have that $G|_T \approx_{2^t \cdot \varepsilon} \text{Unif}(\{0,1\}^t)$, and thus we definitionally show $(\varepsilon \cdot 2^t)$-almost t-wise independence. □

To obtain $(1 - 2^{-t}) \cdot m$ satisfied clauses, we need a δ-almost t-wise independent G such that $\delta \leq \frac{1}{2m}$; per Lemma 10, this means we need to find an ε-biased G such that $\varepsilon \leq \frac{1}{2^t \cdot 2m}$, so we are now on the hunt for how to create such a generator.

Definition 11 (ε-balance)**.** A $[N, k, d]_2$-linear code C is ε-balanced if for all $x, y \in C$, $\Delta(x, y) \in \left(\frac{1}{2} \pm \varepsilon\right) N$

Lemma 12. *If $m \in \mathbb{F}_2^{k \times N}$ is the generator of an ε-balanced code, then the function $G : [N] \to \mathbb{F}_2^k$ which given a number $i \in [N]$ outputs the i-th column of m is ε-biased.*

Proof. We note that per the definition of ε-balanced, any column in m has a weight within $\left(\frac{1}{2} \pm \varepsilon\right) N$. As such, we see that for any linear function $L : \mathbb{F}_2^k \to \mathbb{F}_2$, $x \sim \text{Unif}(\{0,1\}^k)$, and $z \sim \text{Unif}([N])$, $E[L(G(z))]$ is within ε of $E[L(x)]$, thus completing the proof. □

Now, we specifically want a larger k (since $k = n$ for our initial representation of G, and thus would indicate a longer output), a smaller N (which would yield a smaller seed length), and $d \in \left(\frac{1}{2} \pm \varepsilon\right) N$. This is equivalent to searching for a code with good rate, which has been something we've done this whole semester!

Specifically, we note that we have found linear codes of form $N = \frac{k^2}{\varepsilon^2}$ (RSHD) and $N = \frac{k}{\varepsilon^3}$ (AGHD) from Problem Set 3.

Exercise 13. *Prove that the codes we found with bounds $N = \frac{k}{\varepsilon^3}$ and $N = \frac{k^2}{\varepsilon^3}$ above contain ε-balanced subcodes. A suggested approach is to consider separating the codes into two disjoint cosets, one which will form the subcode and the other which contains elements close to* 1.

Now, using $k = n$, $\varepsilon = \frac{1}{2^t \cdot 2m}$, we have that

$$N = 2^S = O(2^{2t} m^4),$$

meaning the overall runtime of the deterministic algorithm will be $O(2^{2t} m^5)$. But can we do better?

Lemma 14. For ε-biased $G_1 : \{0,1\}^S \to \{0,1\}^a$ and t-wise independent and linear $G_2 : \{0,1\}^a \to \{0,1\}^n$, $G = G_2 \circ G_1 : \{0,1\}^S \to \{0,1\}^n$ is $(\varepsilon \cdot 2^t)$-almost t-wise independent.

Proof. For any w which is distributed on $\{0,1\}^a$ and $T \subseteq [n]$ such that $|T| \leq t$, $G_2(w)|_T$ will have the same parities of w since G_2 is linear and t-wise independent. Then, we note that $L \circ G_2$ is linear since the composition of linear functions is linear, and thus we can treat $L \circ G_2$ as a linear test, where specifically if $L \neq 0$, $L \circ G_2|_T \neq 0$ since G_2 is t-wise independent. Then, we note that G_1 being ε-biased means it ε-fools any linear test, meaning that by extension $(G_2 \circ G)|_T$ is ε-biased and thus is $(\varepsilon \cdot 2^t)$-almost t-wise independent by Lemma 10. □

Now, harkening back to the BCH codes we used, we want $a = \frac{t}{2} \log n$. This yields $2^S = \frac{a^2 2^{2t}}{\varepsilon^2} = \frac{t^2 \log^2(n) 2^{2t}}{4\varepsilon^2}$ and thus a runtime of $O(\frac{2^{2t}}{\varepsilon^2} \cdot \log^2(n) \cdot n)$ which is $(\varepsilon \cdot 2^t)$-almost t-wise independent by Lemma 14 and thus per the findings of Exercise 8 is $(1 - 2^{-t} - \varepsilon)$-approximator for max t-SAT.

3 Proof Sketches for the Exercises

Proof sketch for Exercise 2. We can generalize Johnson's algorithm for MAX-3SAT to MAX t-SAT: specifically, the algorithm will randomly assign the boolean values per our random Bernoulli assignment algorithm, and continue until equal or more than $(1 - 2^{-}t)$ m clauses are satisfied. First, there is a well-defined answer for this since we note that the expected number of satisfied clauses from our random Bernoulli assignment algorithm is $(1 - 2^{-}t)m$, meaning that there are assignments which can be outputted by our algorithm. Then, we note that since there is a 1/2 probability for outputting an assignment which satisfies equal to or more than the expected value, we see that in setting a limit of n rounds before our algorithm halts should it not have found an appropriate assignment, the probability of failure is $2^{-}n$, which is exponentially small in n. □

Proof sketch for Exercise 8. From our definition of delta-almost t-wise independent spaces, we see that the maximum probability of error for the assignment of t variables is $(1/2^t)(1 - \delta)$ since it can be at most delta away from the uniform value of $1/2^t$. Thus, the overall expected number of fulfilled clauses is $(1 - 1/2^t(1 - delta))m$. □

Proof sketch for Exercise 13. For this, one would first show that the aforementioned codes can be made epsilon-biased, and then proceed to show how to construct an epsilon-balanced code from such epsilon-biased codes. For the conversion, we would note that epsilon biased codes can be made into generating matrices for epsilon-balanced codes. □