CS265, Fall 2023

Class 12: Agenda and Questions

1 Announcements

- HW5 due Friday!
- HW6 out now!
- No class on Tuesday, November 7 (Democracy/Election day! Vote if you are eligible!)
- HW7 won't be due until after Fall break—get some extra rest but still come to class!

2 Questions?

Any questions from the minilectures and/or the quiz? (Constructive LLL)

3 You prove the constructive LLL for another problem!

In the mini-lectures we proved the constructive LLL for the special case of SAT. Now we'll do it for another problem. (Of course, it can be done in general, but two examples basically makes a theorem...)

Consider the following problem (which has featured on a quiz). You are coloring the integers $\{1, \ldots, n\}$ either blue or red. You are given as input a collection of sets $S_1, S_2, \ldots, S_m \subseteq \{1, \ldots, n\}$, so that:

- Each set S_i has size at least k.
- Each set S_i intersects at most d other sets S_j, for some $d > 1$.

Our goal is to color the points $\{1, \ldots, n\}$ so that there is no monochromatic set S_i. (A set S_i is *monochromatic* if every element of it is either red or blue).

Group Work

1. Mimic the proof of the constructive LLL that we saw for k-SAT to give a randomized algorithm that does the following.

   Suppose that $k \geq \log_2 d + 10000$. (Here, 10000 is a stand-in for "some big enough constant.") Then there is a randomized algorithm that proceeds by re-randomizing the sets S_i, (that is, it will iteratively look at different sets S_i and randomly re-color all of the points in that set), so that:

- If the algorithm terminates, then all of the numbers $\{1, \ldots, n\}$ will be colored so that there is no monochromatic set S_j.
- The expected number of times that the algorithm re-randomizes a set S_j is $\text{poly}(m)$.

Don't worry about giving a complete proof with all the details, just work it out with enough detail that you believe it. As we did in the minilecture video (Fact 4 of Lecture Notes 12), you may use the (informal) fact that for a random binary string of length X, with high probability it cannot be compressed to fewer than X bits: namely "there is no compression function $f : \{0,1\}^X \to \{0,1\}^Y$ so that (a) $Y \ll X$ and (b) with high probability over a uniformly random $x \in \{0,1\}^X$, it is possible to recover x given $f(x)$."

Hint*: To map this problem onto k-SAT, think of the S_j's as standing in for clauses, and the numbers $\{1, \ldots, n\}$ as standing in for variables.*

Hint*: It's not quite as straightforward as applying the mapping in the previous hint and calling it a day. In particular, can you still work backwards from the "print" statements in the k-SAT version to figure out the original random bits?*

2. What happens to your proof if the number of possible colors grows from two (blue and red) to some number t? In particular, can you get the same guarantee as above, but under a weaker guarantee (eg, $k \geq$ [something smaller than $\log d + 10000$]).
3. How does the answer that you got in the previous part compare to what Corollary 3 in the lecture notes would give you for this problem?

   As a reminder, that Corollary says:

   > Let V be a finite set of independent random variables. Let $\mathcal{A}$ be a finite set of events determined by the random variables in V. If for all $A \in \mathcal{A}$, $|\Gamma(A)| \leq d+1$, and $\Pr[A] \leq \frac{1}{e(d+1)}$, then the algorithm from the lecture notes (the general one, not just for k-SAT) will find an assignment to the variables V such that no event of $\mathcal{A}$ occurs. Additionally, the expected number of "re-randomizations" performed by the algorithm is bounded by $O(|\mathcal{A}|/(d+1))$.

4. [**This question is open-ended and may be difficult—think about it after you finish the others if you still have time.**] What happens to your proof if the sets can have variable size? (e.g., if all but a few of them have size k, and a few can be really small? Or if they have average size k? Or....?)