CS 761: Randomized Algorithms Fall 2019

Lecture 11 — November 22, 2019

Prof. Gautam Kamath *By: Yossef Musleh, Ali Sabet*
Edited by Vedat Levi Alev

Disclaimer: These notes have not been subject to the usual scrutiny reserved for formal publications.

Property Testing

A Motivating Example

Suppose Alice and Bob each have n-bit strings $a = a_1 \dots a_n$ and $b = b_1 \dots b_n$ respectively, and they want to test if $a = b$ while minimizing the number of bits communicated. It turns out that this can be done

- Deterministic: $\Theta(n)$ bits
- Randomized: $\Theta(\log n)$ bits
- Property Testing: $O(1/\epsilon)$ bits

The property testing approach relaxes the problem by assuming that either $a = b$ or they are at least "ϵ-far" away i.e. $\|a - b\|_1 \geq \epsilon n$. Under this assumption, consider the following algorithm:

- Alice picks $O\left(\frac{\log 1/\delta}{\epsilon}\right)$ indices using a shared source of randomness, and sends her corresponding bits to Bob.
- If $a_i = b_i$ for all shared indices i, return $a = b$, otherwise return $a \neq b$

If $a = b$, the answer is always correct, but what happens when a and b are far apart? In this case we have

$$\Pr\left[\text{return } a = b \mid \|a - b\|_1 \geq \epsilon n\right] \leq \left(1 - \frac{\epsilon n}{n}\right)^{O(\log(1/\delta)/\epsilon)} \leq (\exp(-\epsilon))^{O(\log(1/\delta)/\epsilon)} \leq \delta.$$

Theorem 1. *There exists a $O\left(\frac{\log 1/\delta}{\epsilon}\right)$-bit communication protocol to test whether two n-bit strings a, b satisfy either $a = b$ or $\|a - b\| \geq \epsilon n$ with probability $1 - \delta$.*

Testing Sortedness

Given an array x of n (unique) values, test whether they're sorted or if they are "ϵ-far," i.e., we must remove $\geq \epsilon n$ values to make the rest sorted. The array

$$[1, 2, \underbrace{6, 5}_{\text{remove one}}, 7, 8]$$

can be made sorted by removing one element and hence is not ϵ-far.

We first investigate a few natural algorithms, and show why they are query-inefficient for this problem.

Algorithm 1: Pick a random index $i \in [n]$ and check if $x_i < x_{i+1}$.

Problem: If there are lots of contiguous sorted blocks, if the blocks themselves are not in sorted order with respect to one another, we only return "not sorted" if we pick boundary elements. For example:

$$\frac{3n}{4}+1, \ldots, n, \frac{n}{2}+1, \ldots, \frac{3n}{4}, \frac{n}{4}+1, \ldots, \frac{n}{2}, 1, \ldots, \frac{n}{2}$$

Algorithm 2: Pick 2 random indices $i < j$ and check if $x_i < x_j$.

Problem: If there are small unsorted blocks but the blocks are in relatively sorted order, we only fail if we choose two elements from the same block. For example:

$$[4, 3, 2, 1], [8, 7, 6, 5], [12, 11, 10, 9], \ldots$$

For the preceding two cases, we need $\Omega(n)$ queries.

Algorithm 3: Pick a random subsequence and test to see if the subsequence is sorted.

Problem: Requires $\Omega(\sqrt{n})$ queries. Consider the case where we have $\sqrt{n}$ subsequences in reverse sorted order of length $\sqrt{n}$ placed in relatively sorted order, as in the following example:

$$[4, 3, 2, 1], [8, 7, 6, 5], [12, 11, 10, 9], [16, 15, 14, 13]$$

We find a subsequence to be out of order only if we compare two entries from the same subsequence, which on average happens after $O(\sqrt{n})$ queries.

Algorithm 4: Repeat $O\big(\frac{\log(1/\delta)}{\epsilon}\big)$ times:

- Pick a random index $i \in [n]$.
- Binary search for x_i.
- If x_i is not found via binary search, return *not sorted*.

Otherwise return *sorted*.

Analysis: Each binary search uses $O(\log n)$ queries, so the overall query complexity is $O\big(\frac{\log(1/\delta)\log n}{\epsilon}\big)$. If sorted, the algorithm always returns the correct answer. If the array is ϵ-far from sorted, we want to show there are many *non-searchable* elements, which are elements where binary search does not correctly return the location of the searched element.

Claim 2. *If an array is ϵ-far from sorted, then there are at least ϵn non-searchable elements.*

Proof. We prove this via contrapositive. Suppose there are less than ϵn non-searchable elements and consider two searchable elements $x_i < x_j$. If x_p is the lowest common pivot, then we must have $x_i \le x_p \le x_j$ so the triple of elements are in sorted order. If there are more than $(1-\epsilon)n$ searchable elements, the previous argument implies that the sub-array of searchable elements is in sorted order and so fewer than ϵn elements need to be removed. □

So then we have

$$\Pr\big[\text{finding a searchable element}\big] \le \left(1 - \frac{\epsilon n}{n}\right)^{O(\log(1/\delta)/\epsilon)} \le \big(\exp\big(-\epsilon\big)\big)^{O(\log(1/\delta)/\epsilon)} \le \delta.$$

Aside: Graphs and functions are two common types of objects to which property testing approaches can be applied. There are a number properties commonly considered when testing graphs, including whether a graph is bipartite, planar, or triangle-free. These can also be tested under various assumptions, such as the graph being dense or having a degree bound. In the case of functions, we trade sortedness for monotonicity.

Testing Distributions

Suppose we can sample from a distribution $X_1, \ldots, X_m \sim P$; we might want to test:

- Identity: Whether $P = Q$ for some distribution Q, or $\|P - Q\|_1 \ge \epsilon$. Commonly, $Q = U_n$, the uniform distribution.
- Membership: Whether P is a member of a class of distributions M, or $\min_{Q \in M} \|P - Q\|_1 \ge \epsilon$. For example, M may be the set of monotone distributions.

Uniformity Testing

Suppose either $P = U_n$ or $\|P - U_n\|_1 \ge \epsilon$. How many samples can we expect to need to distinguish the two case with probability $1-\delta$? Intuitively, one might expect $\Omega(\sqrt{n})$; for example, consider the distribution that is uniform over a subset of size $n/2$ and zero elsewhere. We don't expect to get any information until we begin to see collisions in the sample set, which requires $\Omega(\sqrt{n})$ samples.

More formally, the probability that any two samples are equal is

$$\sum_{i=1}^{n} p_i^2 = \|P\|_2^2.$$

If $P = U_n$, then $\|P\|_2^2 = \frac{1}{n}$. Moreover, if $\|P - U_n\|_1 \geq \epsilon$ by Cauchy-Schwarz we have:

$$\epsilon \leq \|P - U_n\|_1 \leq \sqrt{n}\|P - U_n\|_2.$$

So then we get:

$$\frac{\epsilon^2}{n} \leq \|P - U_n\|_2^2 = \sum_{i=1}^{n} \left(p_i - \frac{1}{n}\right)^2 = \sum_{i=1}^{n} p_i^2 - \frac{2}{n}\sum_{i=1}^{n} p_i + \frac{1}{n} = \|P\|_2^2 - \frac{1}{n}.$$

So $\|P\|_2^2 \geq \frac{1}{n} + \frac{\epsilon^2}{n}$, and it suffices to obtain an (ϵ^2, δ) approximation of $\|P\|_2^2$. To do this, consider the following statistic:

$$Z = \frac{1}{\binom{m}{2}} \sum_{1 \leq i < j \leq m} \sigma_{ij}$$

where

$$\sigma_{ij} = \begin{cases} 1, & \text{if } X_i = X_j \\ 0, & \text{otherwise.} \end{cases}$$

Then:

$$E[Z] = \frac{1}{\binom{m}{2}} \binom{m}{2} E[\sigma_{12}] = \Pr[X_1 = X_2] = \|P\|_2^2$$

$$E[Z^2] = E\left[\left(\frac{1}{\binom{m}{2}} \sum_{1 \leq i < j \leq m} \sigma_{ij}\right)^2\right] = \frac{1}{\binom{m}{2}^2} \sum_{\substack{1 \leq i < j \leq m \\ 1 \leq k < \ell \leq m}} E[\sigma_{ij}\sigma_{jk}]$$

Consider the following cases:

Case 1 $i = k$, $j = \ell$:

$$E[\sigma_{ij}^2] = E[\sigma_{ij}] = \Pr[X_i = X_j] = \|P\|^2$$

$$\sum_{\substack{i=k \\ j=\ell}} E[\sigma_{ij}\sigma_{k\ell}] = \binom{m}{2}\|P\|_2^2$$

Case 2 $3/4$ of i, j, k, ℓ are unique:

$$\sum_{\substack{i=k \\ \text{or } j=k \\ \text{or } i=\ell \\ \text{or } k=\ell}} E[\sigma_{ij}\sigma_{k\ell}] = \binom{m}{3} 4\|P\|_3^3 \leq 4\binom{m}{3}\|P\|_2^3$$

Case 3 All unique - this implies $\sigma_{ij}, \sigma_{k\ell}$ are independent:

$$\sum_{\substack{i,j,k,\ell \\ \text{distinct}}} E[\sigma_{ij}\sigma_{k\ell}] = \sum_{\substack{i,j,k,\ell \\ \text{distinct}}} E[\sigma_{ij}]^2 = \binom{m}{2}\binom{m-2}{2}\|P\|_2^4$$

Combining the cases:

$$Var[Z] \leq \frac{1}{\binom{m}{2}^2}\left(\binom{m}{2}\|P\|_2^2 + \binom{m}{3}\|P\|_2^3 + \underbrace{\binom{m}{2}\binom{m-2}{2}\|P\|_2^4 - \binom{m}{2}^2\|P\|_2^4}_{<0}\right)$$

$$\leq \frac{1}{\binom{m}{2}^2}\left(\binom{m}{2}\|P\|_2^2 + \binom{m}{3}\|P\|_2^3\right).$$

By Chebyshev:

$$\Pr\left[|Z - \|P\|_2^2 \geq \epsilon^2\|P\|_2^2\right] \leq \frac{Var[Z]}{(\epsilon^4\|P\|_2^4)}$$

$$\leq \frac{1}{\epsilon^4\|P\|_2^4}\left(\frac{\|P\|_2^2}{m^2} + \frac{\|P\|_2^3}{m}\right) \leq \frac{n}{\epsilon^4 m^2} + \frac{n^{1/2}}{\epsilon^4 m}.$$

So we need $m \geq \Omega(\sqrt{n}/\epsilon^4)$. This implies the following algorithm:

- Take $\Omega(\sqrt{n}/\epsilon^4)$ samples from a distribution P
- Compute Z
- If $Z \leq \frac{1}{n} + \frac{\epsilon^2}{2n}$ output *uniform* otherwise *not uniform*

The analysis is lossy - the same test actually achieves sample complexity $O\left(\frac{\sqrt{n}}{\epsilon^2}\right)$ [4]. Furthermore, requiring equality is strong; instead we may wish to distinguish between $\|P - U_n\| \leq \frac{\epsilon}{2}$ or $\|P - U_n\| \geq \epsilon$. But this turns out to have sample complexity $\Omega\left(\frac{n}{\log n}\right)$ [5]! One can note that the problem of distinguishing $\|P - U_n\|_1 \leq O\left(\frac{\epsilon^2}{100\sqrt{n}}\right)$ versus $\|P - U_n\| \geq \epsilon$ can be done with only $\Omega(\frac{\sqrt{n}}{\epsilon^2})$ samples. This is because U_n and any P such that $\|P - U_n\|_1 \leq O(\tau)$ can be coupled to generate an identical set of n samples with high constant probability, for any $n \leq O(1/\tau)$.

We instead relax the tolerance of the tester into the χ^2 distance, and show that this testing procedure can be done with the same sample complexity. But is this relaxed tolerance into Chi-squared still useful? Yes, and the method for monotonicity testing is the key application [1].

Definition 3. *Let P, Q be two distributions. Define*

$$\chi^2(P, Q) = \sum_{i \in S} \frac{(p_i - q_i)^2}{q_i}.$$

Observe that $\chi^2(P,Q)$ can be large even when $\|P-Q\|_1 \approx 0$, such as when $P = Ber(1)$, $Q = Ber(1-\gamma)$. As $\gamma \to 0$, $\|P-Q\|_1 \to 0$, but $\chi^2(P,Q) \to \infty$. However, we always have $\|P-Q\|_1 \leq \chi^2(P,Q)$.

Learning the Test for Monotonicity

To motivate the need for χ^2-distance, we first give a naive algorithm for testing whether a distribution is monotone, which will have a high sample complexity. We then discuss the modifications needed to achieve the optimal sample complexity.

1. Learn P, assuming P is monotone.
   (a) If P is monotone, get $\hat{P}$ such that $\|P-\hat{P}\|_1 \leq \frac{\epsilon}{100}$
   (b) If $\|P-M_n\|_1 \geq \epsilon$, get $\hat{P}$ such that $\|P-\hat{P}\|_1 \geq \epsilon$
2. Test whether
   (a) If $\|P-\hat{P}\|_1 \leq \frac{\epsilon}{100}$, then return *monotone*.
   (b) If $\|P-\hat{P}\|_1 \geq \epsilon$, then *not monotone*.

The first step can be done with $O(\frac{\log n}{\epsilon^3})$ samples using domain compression [3], whereas the second takes $\Omega(\frac{n}{\log n})$ samples, which is not great.

So instead, we switch from ℓ_1 learning to χ^2 learning by replacing the condition $\|P-\hat{P}\|_1 \leq \frac{\epsilon}{100}$ with $\chi^2(P,\hat{P}) \leq \frac{\epsilon^2}{100}$. This turns out to have a sample complexity of $O(\frac{\log n}{\epsilon^4})$ [1, 2]. The increase in sample complexity is modest: a multiplicative $O(1/\epsilon)$, compared with before.

What about the second step? Now we must solve a more difficult testing problem than before, which tests whether a distribution is close in χ^2-distance or far in ℓ_1-distance from a known distribution. Fortunately, this can still be solved with $O(\sqrt{n}/\epsilon^2)$ samples.

Theorem 4. *For a known distribution Q, there exists an algorithm with sample complexity $O(\frac{\sqrt{n}}{\epsilon^2})$ which distinguishes between the cases $\chi^2(P,Q) < \epsilon^2/10$ versus $\|P-Q\|_1 > \epsilon$ with probability at least $5/6$.*

The statistic is the following, which is similar to Pearson's classical Chi-squared statistic.

$$Z(Q) = \sum_{i \in [n]} \frac{(N_i - mq_i)^2 - N_i}{mq_i}$$

where

- N_i = number of samples equal to i
- m = total number of samples
- q_i = Probability of i under Q

Putting these two steps together, the overall sample complexity is $O(\frac{\sqrt{n}}{\epsilon^2} + \frac{\log n}{\epsilon^4})$.

References

[1] Jayadev Acharya, Constantinos Daskalakis, and Gautam Kamath. Optimal testing for properties of distributions, 2015.

[2] Jayadev Acharya, Ashkan Jafarpour, Alon Orlitsky, and Ananda Theertha Suresh. Efficient compression of monotone and m-modal distributions. In *2014 IEEE International Symposium on Information Theory*, pages 1867–1871. IEEE, 2014.

[3] Lucien Birgé et al. Estimating a density under order restrictions: Nonasymptotic minimax risk. *The Annals of Statistics*, 15(3):995–1012, 1987.

[4] Ilias Diakonikolas, Themis Gouleakis, John Peebles, and Eric Price. Collision-based testers are optimal for uniformity and closeness. *arXiv preprint arXiv:1611.03579*, 2016.

[5] Gregory Valiant and Paul Valiant. Estimating the unseen: an n/log (n)-sample estimator for entropy and support size, shown optimal via new clts. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 685–694. ACM, 2011.