CS6025 Report
Testing Monotonocity

Amit Roy

May 8, 2019

Contents

1 Introduction

In the class we have seen Montonocity Testing(randomised) for boolean functions. We will further extend monotonocity testing for general functions $f : \Sigma^n \to \mathcal{Z}$ where Σ is a finite alphabet and $\mathcal{Z}$ is finite range. We will also look on Unateness testing of a function $f : \{0,1\}^n \to \{0,1\}$.

2 Monotonocity

A function $f : \Sigma^n \to \mathcal{Z}$ is monotone if $f(x) \leq_{\mathcal{Z}} f(y)$ for every $x \prec_{\mathcal{Z}} y$. We will first show monotonocity testing for $f : \Sigma^n \to \{0,1\}$ where $|\Sigma| > 2$ and later generalise it to any $\mathcal{Z}$.
Algorithm 1:

1. Uniform select $i \in \{1 \ldots, n\}, \alpha \in \Sigma^{i-1}$ and $\beta \in \Sigma^{n-i}$
2. Select (k, l) according to distribution p.
3. If $f(\alpha k \beta) > f(\alpha l \beta)$ then *reject*

2.1 Idea

The analysis of the above algorithm entirely reduces to the analysis of the algorithm for $n = 1$ i.e. $g : \Sigma \to \{0,1\}$. We will show that error in one iteration of the algorithm given above is upper bounded by performance of a function g. So bounding the error for the function will bound the error for $f : \Sigma^n \to \{0,1\}$. So we will see different modifications of the algorithm for $n = 1$ and their performance will improve the main algorithm.

2.2 Lemmas and Propositions

Lemma 1. *Let A denotes a single iteration of the algorithm given above and $f : \Sigma^n \to \{0,1\}$, then there exists a function $f_{i,\alpha,\beta} : \Sigma \to \{0,1\}$ such that the follwing holds*

1. $\epsilon_M(f) \leq 2\Sigma_i \mathbb{E}[\epsilon_M(f_{i,\alpha,\beta})]$
2. $\mathbb{E}_{i,\alpha,\beta}[Pr[A \ rejects \ f_{i,\alpha,\beta}]] \leq Pr[A \ rejects \ f]$

Hence we will focus on designing algorithms for the case $n = 1$. Note that we mentioned about the distribution p in the algorithm. We will analyse now for three different distributions and see the query complexity.
Algorithm 1.1 This is a modification of the algorithm we have given in the beginning. The distribution here is $p_1 : \Sigma \times \Sigma \to [0,1]$ defined by $p_1(k, k+1) = \frac{1}{d-1}$ for $k = 1, \ldots d-1$.

Proposition 1 Let A_1 be a single iteration of the Algorithm 1 and $f' : \Sigma \to \{0,1\}$. Then , $Pr[A_1 \ rejects \ f'] \geq \frac{2}{d-1}\epsilon_M(f')$

Algorithm 1.2 The distribution used in this algorithm is $p_2 : \Sigma \times \Sigma \to [0,1]$ which is uniform on the set P defined as follows

$$P = \{(k,l) \mid 0 < l - k \leq 2^t \text{ and } 2^t \text{ is the largest power of 2 which divides either k or l}\}$$

Proposition 2 Let A_2 be the single iteration of the Algorithm 1.2 and $f' : \Sigma \to \{0,1\}$. Then , $Pr[A_2 \ rejects \ f'] \geq \Omega(\frac{1}{\log d})\epsilon_M(f')$

Algorithm 1.3 The distribution used in this algorithm is uniformly chosen from the set P which is defines as follows.

$$P = \{(k,l) \mid 1 \leq k < l \leq d\}$$

Let $p_3 : \Sigma \times \Sigma \to [0,1]$ defined by $p_3(k,l) = \frac{2}{d(d-1)}$

2.3 Main Theorem

Theorem 1. *[GGL+00]*

1. *The query complexity of Algorithm 1.1 is* $\mathcal{O}(\frac{nd}{\epsilon})$
2. *The query complexity of Algorithm 1.2 is* $\mathcal{O}(\frac{n \log d}{\epsilon})$
3. *The query complexity of Algorithm 1.3 is* $\mathcal{O}(\frac{n}{\epsilon})^2$

This follows from Lemma 1 and the propositions for each of the algorithms.

2.4 General Ranges

We will lose a factor of $|\mathcal{Z}|$ while extending to any ranges. The same algorithm will work. Let $f : \Sigma^n \to \{0,1,\ldots b\}$ and $f_i : \Sigma^n \to \{0,1\}$ where $f_i(x) = 1$ if $f(x) \geq i$ for $i = 0,1,\ldots,b$. Let $\delta_M^A(f)$ be the probability that the algorithm A observes a violating edge pair in f and $\epsilon_M(f)$ be the distance from the closest monotone function.

Lemma 2. *Let* $f : \Sigma^n \to \{0,1\ldots,b\}$ *and* f_i *as defined above*

1. $\epsilon_M(f) \leq \Sigma_{i=1}^b \epsilon_M(f_i)$
2. $\delta_M^A(f) \geq \delta_M^A(f_i)$, *for every* i

We know that $\delta_M^A(f_i) \geq \epsilon_M(f_i)/F$ where F depends on Σ and n. From above lemma we get that

$$\delta_M^A(f) \geq max\{\delta_M^A(f_i)\} \geq \frac{1}{b}\Sigma_{i=1}^b \delta_M^A(f_i) \geq \frac{1}{b}\Sigma_{i=1}^b \frac{\epsilon(f_i)}{F} \geq \frac{1}{b}\frac{\epsilon_M(f)}{F}$$

2.5 Recent Development

In 2013, Chakrabarty and Seshadhri [CS13] removed the dependence on the range size showing an $\mathcal{O}(\frac{d \log n}{\epsilon})$- query monotonocity tester for any real valued function and then again in a different paper [CS13] they showe d that any monotonocity tester(adaptive) with distance parameter ϵ must make $\Omega(\frac{d \log n}{\epsilon})$ queries.

3 Unateness

We will define unateness first. If $f(x_1, \ldots x_{i-1}, 1, x_{i+1} \ldots x_n)/gef(x_1, \ldots x_{i-1}, 0, x_{i+1} \ldots x_n)$ then f is called to be positive unate and if $f(x_1, \ldots x_{i-1}, 1, x_{i+1} \ldots x_n)/gef(x_1, \ldots x_{i-1}, 0, x_{i+1} \ldots x_n)$ then f is called to be negative unate. A function f is called to be unate if for every x_i , it is either positive or negative unate.

Algorithm

1. Sample $m = \mathcal{O}(\frac{n^{1.5}}{\epsilon})$ strings uniformly and m indices
2. From the chosen string x^j , obtain y^j by flipping one bit and check unateness.

3.1 Idea

If the function f is unate then the algorithm never makes mistake. We only need to analyse for the false positive. The authors tries to prove that if f is ϵ-far from being unate then the algorithm will reject with probability at least ϵ and end up showing that $\delta_M(f) \geq \frac{\epsilon_M(f)}{n}$

3.2 Main Theorem

Theorem 2. *[GGL[+]00] The algorithm given is a testing algorithm for unateness and if the function is unate then algorithm always accepts.*

Some more definitions

$$\overline{\pi} = \pi_1 \ldots \pi_n \text{ where } \pi \text{ is a permutation } \{0,1\} \; or \; \{1,0\}$$
$$x <_{\overline{\pi}} y \text{ iff } x_i <_{\pi_i} y_i$$
$$\epsilon_{M,\overline{\pi}}(f) = \text{ Min distance between } f \text{ and any function } g \text{ that is mononotone with respect to } \pi$$
$$\delta_{M,\overline{\pi}} = x <_{\overline{\pi}} y \text{ but } f(x) > f(y) \text{ and the pair } (x,y) \text{ differs only in 1 bit}$$
$$\epsilon_U(f) = \text{ minimum distance of } f \text{ from set of unate functions}$$
$$\gamma_{i,\pi}(f) = \text{ fraction of strings that differ on single bit (i) and } x_i <_{\pi} y_i \text{ but } f(x) > f(y)$$

Lemma 3. *For any function $f : \{0,1\}^n \to \{0,1\}$ and a permutation $\overline{\pi}$*

$$\delta_{M,\overline{\pi}}(f) \geq \frac{\epsilon_{M,\overline{\pi}}}{n}$$

Now we need to relate $\delta_{M,\overline{\pi}}(f)$ and behaviour of the given algorithm for unateness. The key observation in this follows from the following eqaution

$$\delta_{M,\overline{\pi}}(f) = \Sigma_{i=1}^n \gamma_{i,\pi_i}(f) = \Sigma_{i=1}^n min_{\pi}\{\gamma_{i,\pi_i}(f)\}$$

This is clear from the definition of each of terms in the equation.

Lemma 4. $\Sigma_{i=1}^n min_{\pi}\{\gamma_{i,\pi}\} \geq \frac{\epsilon_U(f)}{n}$

This follows from $\Sigma_{i=1}^n min_{\pi}\{\gamma_{i,\pi}\} = \delta_{M,\overline{\pi}}(f) \geq \frac{\epsilon_{M,\overline{\pi}}}{n} \geq \frac{\epsilon_U(f)}{n}$

References

[CS13] D Chakrabarty and C Seshadhri. Testing monotonicity. *STOC*, 2013.

[GGL+00] Oded Goldreich, Shafi Goldwasser, Eric Lehman, Dana Ron, and Alex Samorodnitsky. Testing monotonicity. *Combinatorica*, 20(3):301–337, Mar 2000.