6.896 Sublinear Time Algorithms	November 30, 2004
Lecture 21	
Lecturer: Eli Ben-Sasson	*Scribe: Mihai Pătraşcu*

1 Testing Monotonicity

Previously, we have seen how to test the linearity of a function. We will now consider another interesting property of a function, namely monotonicity. There have been several results on testing monotonicity; the one we present is from Goldreich et al [1].

In Lecture 1 we have already seen monotonicity testing in one dimension, so now we concentrate on functions with several variables. We will only consider the canonical multidimensional case, the Hamming cube, though this can be generalized.

Definition 1 (Distance) *The distance between two functions f and g is defined as $\delta(f,g) = \Pr_x[f(x) \neq g(x)]$, where x is drawn uniformly from the domain. The distance between f and a set of functions S is $\delta(f,S) = \min_{g \in S} \delta(f,g)$.*

Definition 2 (Monotonicity) *For $x, y \in \{0,1\}^n$, say $x \preceq y$ if $(\forall) i \in [n], x_i \leq y_i$. A function $f : \{0,1\}^n \to \{0,1\}$ is monotone iff $(\forall) x, y \in \{0,1\}^n$, $x \preceq y$ implies $f(x) \leq f(y)$. Let M_n be the set of monotone boolean functions on n variables.*

Given n and oracle access to a function $f : \{0,1\}^n \to \{0,1\}$, our problem is to test proximity to M_n (in a sense that will be defined below). The tester is quite simple, following the spirit of our linearity test.

Monotonicity Tester: Pick $x \in \{0,1\}^n$ and $i \in [n]$ uniformly (and independently) at random. Let y be x with the i-th bit reversed. Accept iff $f(\min\{x,y\}) \leq f(\max\{x,y\})$.

Theorem 3 *Let $\varepsilon(f)$ to be the probability that the monotonicity tester rejects f. Then for every Boolean function $f : \{0,1\}^n \to \{0,1\}$,*

$$2\delta(f, M_n) \geq \varepsilon(f) \geq \frac{\delta(f, M_n)}{n}$$

In this lecture, we will only prove the lower bound on $\varepsilon(f)$. It can be shown that both inequalities are tight for this algorithm [1]. The lower bound may seem small due to the n factor, but one should remember that n is the logarithm of the domain size (a function truth table, of size 2^n). Before proving the Theorem, let us show it implies a tester for M_n in the usual sense.

Corollary 4 *For every integer n, the property M_n is testable with query complexity $O(n)$. Namely, given distance parameter $\delta_0 > 0$, there exists a test that makes $O(n/\delta_0)$ queries, accepts with probability one every $f \in M_n$, and rejects with probability $\geq 2/3$ any function such that $\delta(f, M_n) \geq \delta_0$.*

Proof Repeat the above mentioned monotonicity test $O(n/\delta_0)$ times, and accept iff all repetitions accept. The query complexity is as claimed (because a single test makes only two queries). If $f \in M_n$, the test always accepts. If, on the other hand, $\delta(f, M_n) \geq \delta_0$, a single repetition rejects with probability at least $p = \delta_0/n$. Repeating the test $O(1/p)$ times boosts the rejection probability to $2/3$. ■

The fact that we have both lower and upper bounds on $\varepsilon(f)$ has an interesting consequence, that relates to the recent trend of *tolerant* property testing. A tolerant tester for a property should have rejection probability that is in correlation to the distance of the input from the property. In other words, whereas a (standard) property tester may reject some inputs that are close to the property, a tolerant tester can not do this, rather it has to accept with high probability inputs that are close to being good. Theorem 3 implies a pair of tests, one for rejecting inputs that are far from monotone (as described in the proof of Corollary 4), and another for ascertaining that a function is close to monotone.

2 Proof of Theorem 3

For $i \in [n]$, we say that f is monotone in dimension i if

$$(\forall)\alpha \in \{0,1\}^{i-1}, \beta \in \{0,1\}^{n-i}, f(\alpha 0 \beta) \leq f(\alpha 1 \beta).$$

For a set T, we say f is monotone in dimensions T if it is monotone in every dimension $i \in T$. The key observation of our proof is that if f is monotone in dimensions T, then correcting it to be monotone in dimension $i \notin T$ will not ruin the monotonicity in T.

If f is not monotone in dimension i, we want to correct this. Let the switch of f on the i-th coordinate be the function $S_i(f) : \{0,1\}^n \to \{0,1\}$ defined by the following rule. For all $\alpha \in \{0,1\}^{i-1}, \beta \in \{0,1\}^{n-i}$, if $f(\alpha 0 \beta) > f(\alpha 1 \beta)$, then $S_i(f)(\alpha 0 \beta) = f(\alpha 1 \beta)$ and $S_i(f)(\alpha 1 \beta) = f(\alpha 0 \beta)$. In all places where f does not violate monotonicity in dimension i, $S_i(f)$ is the same as f.

Let U be the set of pairs that can be queried by the monotonicity tester, namely

$$U = \{(x,y) \mid x, y \text{ differ in one bit}, x \preceq y\}.$$

Observe that $|U| = \frac{1}{2} \cdot 2^n \cdot n$. Let $B(f) \subset U$ be the set of all inputs that cause the monotonicity tester to reject, namely

$$B(f) = \{(x,y) \in U \mid f(x) > f(y)\}.$$

Now observe that $\varepsilon(f) = \frac{|B(f)|}{|U|}$. Finally, let

$$D_i(f) = |\{x \mid S_i(f)(x) \neq f(x)\}|.$$

Since $S_i(f)$ corrects counterexamples on the i-th coordinate, we have:

$$\sum_{i=1}^{n} D_i(f) = 2|B(f)| = 2^n \cdot \varepsilon(f) \cdot n$$

Our result will follow directly from the pair of Lemmas below.

Lemma 5 *If f is monotone in dimensions T, then $S_j(f)$ is monotone in dimensions $T \cup \{j\}$.*

Lemma 6 *For any distinct $i, j \in [n]$, we have $D_j(S_i(f)) \leq D_j(f)$.*

We now argue that the Lemmas imply Theorem 3. For ease of notation, let

$$S^{(j)}(f) = S_j(S_{j-1}(\ldots(S_1(f))\ldots)).$$

Define $g = S^{(n)}(f)$, i.e. the completely fixed version of f. Then by lemma 5, g is monotone. By construction, the number of inputs on which f and g differ is precisely $\sum_{i=1}^{n} D_i(S^{(i-1)}(f))$ (where $S^{(0)}(f) \equiv f$). Finally, by Lemma 6, we have $D_i(S^{(i-1)}(f)) \leq D_i(S^{(i-2)}(f)) \leq \ldots \leq D_i(f)$. Therefore:

$$\delta(f, M_n) \leq \delta(f, g) \leq 2^{-n} \sum_{i=1}^{n} D_i(S^{(i-1)}(f)) \leq 2^{-n} \sum_{i} D_i(f) \leq n \cdot \varepsilon(f)$$

This completes the proof of Theorem 3.

3 Proof of the Lemmas

First we show Lemma 6 implies Lemma 5. Clearly $S_j(f)$ is monotone in dimension j. By definition, f is monotone in dimension i iff $D_i(f) = 0$. By Lemma 6, for any $j \in T$ we have $D_i(S_j(f)) \leq D_i(f) = 0$, so S_j must be monotone in dimension j, q.e.d.

Now we prove Lemma 6. Fix i and j, $i < j$. For $\alpha \in \{0,1\}^{i-1}$, $\beta \in \{0,1\}^{j-i}$ and $\gamma \in \{0,1\}^{n-j}$, define $f_{\alpha\beta\gamma}$ as f where we fix all coordinates of the input (with the parameter strings) except i and j. Specifically, $f_{\alpha\beta\gamma} : \{0,1\}^2 \to \{0,1\}$ is defined by $f_{\alpha\beta\gamma}(\sigma\tau) = f(\alpha\sigma\beta\tau\gamma)$. By definition:

$$D_i(S_j(f)) = \sum_{\alpha\in\{0,1\}^{i-1},\beta\in\{0,1\}^{j-i},\gamma\in\{0,1\}^{n-j}} D_2(S_1(f_{\alpha\beta\gamma})) \tag{1}$$

Now fix α, β, γ arbitrarily and denote $f' = f_{\alpha\beta\gamma}$. By Equation (1), Lemma 6 reduces to showing $D_2(S_1(f')) \leq D_2(f')$ for any function $f' : \{0,1\}^2 \to \{0,1\}$. I.e. we have to show that for any 2×2 matrix with $\{0,1\}$-entries, if we sort each column in increasing order, then the number of rows that are ordered decreasingly, does not become larger. This can be done by a simple case analysis, of which we'll do only one example. Suppose after ordering columns of the matrix (denoted A), the bottom row is decreasing (i.e. $A_{1,0} > A_{1,1}$). Since all entries are zero/one, we have $A_{1,0} = 1, A_{1,1} = 0$. Since each column is sorted increasingly, we have $A_{0,1} = 0$. Thus, we conclude the entries of the second column are identical, so sorting columns does not change the number of decreasingly ordered rows (it can only change the location of the decreasing row). The other cases are similarly analyzed. This complete the proof of Lemma 6.

References

[1] Oded Goldreich, Shafi Goldwasser, Eric Lehman, Dana Ron, Alex Samorodnitsky: Testing Monotonicity. Combinatorica 20(3): 301-337 (2000). Previously in FOCS'98.