6.896 Quantum Complexity Theory October 7th, 2008

Lecture 10

Lecturer: Scott Aaronson

"Science: We seek out ignorance and try to demolish it."

Last Time: Grover's algorithm and its optimality

- $D(f)$ = deterministic query complexity of Boolean function $f : \{0,1\}^n \to \{0,1\}$
- $Q(f)$ = quantum query complexity (with bounded error)
- Trivially, $Q(f) \leq D(f)$ for all f.
- $D(OR_n) = n$
- $Q(OR_n) = O(\sqrt{n})$ [Grover]
- $Q(OR_n) = \Omega(\sqrt{n})$ [BBBV]
- $Q(PARITY_n) \leq \frac{n}{2}$ [Deutsch-Jozsa]

Can we show $Q(PARITY_n) \geq \frac{n}{2}$? BBBV gives us $\sqrt{n}$ as a lower bound, if we consider the case with only one bit flipped. It turns out that this bound is tight, and we will develop a new method to prove this.

Today, we will continue our study of query complexity, where we are completely ignoring computation time in our analysis. Recall that this allows for an adaptive algorithm which choses its queries based on the results of previous queries.

1 The Polynomial Method

The so-called polynomial method takes questions about quantum lower bounds and reduces them to questions about the degrees of real polynomials. This approach will perhaps be somewhat more palatable to computer scientists who are still at odds with quantum theory. (Though the application is new, the method is old and derives from a book by Minsky, et al.)

Lemma 1 *(Beals, Buhrman, Cleve, Mosca, de Wolf 1998) Let Q be a quantum algorithm that outputs a boolean value. We are interested in the likelihood that it accepts. If Q makes T queries to input bits $x_1, \ldots, x_n$, then Q's acceptance probability can be represented by a polynomial $p(x_1, \ldots, x_n)$ of degree at most $2T$.*

In other words, if you can prove a lower bound on the degree of a real polynomial that behaves properly (high values for accepted inputs, low values for rejections) then T has to be at least the degree of the polynomial divided by 2, and you get a lower bound on the number of quantum queries.

Proof: Claim: every amplitude can be written as a polynomial of degree at most T after T queries.

By induction on T: Initially all amplitudes $\alpha_{i,2}$ are degree-0 polynomials over $x_1, \ldots, x_n$ - in other words, constants. A query increases degree by 1:

$$\sum \alpha_{i,2} |i, 2\rangle \to \sum \alpha_{i,2}(1 - 2x_i) |i, 2\rangle$$

A unitary operation doesn't change the degree because it is a linear transformation, and we get the two from squaring all coefficients to get a probability. For simplicity, we can assume only real numbers, as shown on the problem set.

$$p = \sum_{|i,2\rangle \text{ accepting}} ||\alpha_{i,2}||^2$$

□

Definition 1 *$\widetilde{deg}(f)$ is the minimum degree of a real polynomial p such that*

$$|p(X) - f(X)| \leq \frac{1}{3} \ \forall X \in \{0,1\}^n$$

Notice that:

$$\widetilde{\deg}(f) \leq 2Q(f)$$

$$Q(f) \geq \frac{\widetilde{\deg}(f)}{2}$$

Now we will attempt to find such polynomials.

Remark 1 *We can assume that p is multilinear without loss of generality. In other words, every term is a product of some subset of the variables to the first power, because the inputs are all 0 or 1, so raising to higher powers is meaningless.*

This task still seems complicated, so we'll learn a new trick that reduces n-dimensional objects to 1-dimensional objects.

Lemma 2 *(Minsky-Papert 1968) Let $q(k) = E_{|X|=k}[p(X)]$ (Expected value over all X with Hamming weight k). Then $q(k)$ is itself a polynomial in k, and $\deg(q) \leq \deg(p)$. One polynomial is over the input bits, the other is over the Hamming weight of the input bits. This process is known as symmetrization.*

Proof:

This is a multilinear polynomial, so we can write it as a sum of linear monomials.

$$p(X) = \sum_{S \subseteq \{1,\ldots,n\}} \alpha_s \prod_{i \in S} x_i, \ \ \alpha_s \in \mathbb{R}$$

Use linearity of expectation:

$$E_{|X|=k}[p(X)] = \sum_{|S|\leq \deg(p)} \alpha_s E_{|X|=k}\left[\prod_{i\in S} x_i\right]$$

This now becomes a combinatorics problem, where we are looking for the probability that all the bits in S will be set to 1. Write out the terms and cancel.

$$= \sum_{|S|\leq \deg(p)} \alpha_s \frac{\binom{n-|S|}{k-|S|}}{\binom{n}{k}}$$

$$= \sum_{|S|\leq \deg(p)} \alpha_s \frac{(n-|S|)!}{n!} k(k-1)(k-2)\cdots(k-|S|+1)$$

(Oh look, a polynomial of degree at most the degree of p.)

What can we say about $p(0,\ldots,0) = q(0)$? $0 \leq q(0) \leq \frac{1}{3}$. Also, $\frac{2}{3} \leq q(1) \leq 1$ and $0 \leq q(2) \leq \frac{1}{3}$. If we draw this polynomial (approximately) we can see that its degree is at least n. (Choose your justification: because it reverse direction n times; if you subtract $1/2$ you can count the zeros, etc.) So:

$$n \leq \deg(q) \leq \deg(p) = \widetilde{\deg}(PARITY_n)$$

$$Q(PARITY_n) \geq \frac{\widetilde{\deg}(PARITY_n)}{2} = \frac{n}{2}$$

Thus, our algorithm is optimal, and parity is in a sense maximally hard for a quantum computer. □

A natural next question is whether this technique can be applied to Grover's algorithm. It turns out that this is true, giving us a completely different proof of the optimality of Grover's algorithm.

Notice that we can't use our polynomial-degree inference method here, because the OR function starts out low and goes high and never goes back. This indicates that the case is more subtle here, even though the polynomial doesn't appear to be low degree (staying flat for a while.)

We turn to a result of Markov, from a conversation with Mendeleyev (of periodic table fame) regarding the maximum absolute value of the derivative of a polynomial in a given range.

Lemma 3 *(A.A. Markov 1889)*

$$\max_{0\leq x\leq n} |p'(x)| \leq \deg(p)^2 \left(\frac{\max_{0\leq x\leq n} p(x) - \min_{0\leq x\leq n} p(x)}{n}\right)$$

Proof: We won't prove this here, but you can, using Chebyshev polynomials. □

In other words

$$\deg(p) \geq \sqrt{\frac{n \cdot MaxDeriv}{MaxHeight}}$$

Notice that $q'(x) \geq \frac{1}{3}$ for some $x \in [0,1]$. There is a subtlety here, because we only know the behavior of the polynomial at integer points. (It could do something wacky!) But, it can't go too extreme without making the max derivative larger.

$$\geq \sqrt{\frac{n \cdot \max\{\frac{1}{3}, 2h-1\}}{h}} = \Omega(\sqrt{n})$$

2 Limiting Possible Speedups

We've seen quadratic speedups, but it is natural to ask if there any total boolean function that gives us an exponential speedup by using a quantum algorithm. Factoring isn't a candidate here, because the speedup is conjectured. Period finding isn't either, because it requires that the input be periodic, which entails a promise about the input.

It turns out that the following result holds:

Proposition 4 *(BBCMW): $D(f) = O(Q(f)^6)$ for all f.*

Conjecture 1 *$D(f) = O(Q(f)^2)$. Scott's intuition.*

Indeed, if you're going to get an exponential speedup, you need some sort of promise about this input.

Definition 2 *Given some boolean function $f : \{0,1\}^n \to \{0,1\}$, a certificate for input $X \in \{0,1\}^n$ is some subset of input bits that proves the value of $f(X)$. For the OR function, most inputs have certificate size 1. The input of all zeros has certificate size 0.*

Let $c_x(f)$ be the minimum size (number of bits) of a certificate for X, and let $C(f) = \max_x c_x(f)$ (Check: for OR_n, this is n.)

Observe that $C(f) \leq D(f)$, because we have to uncover a certificate before terminating, otherwise there are still plausible inputs which are accepted or rejected. However, equality does not hold due to a special function:

"OR of ANDs": Represented by a tree of where the output bit (the root) is the OR of $\sqrt{n}$ nodes at the next level, which are in turn the AND of $\sqrt{n}$ leaves of each node, which is each an input bit. Equivalently, we have a $\sqrt{n} \times \sqrt{n}$ matrix with a bit in each position. Here, the question is whether there is an all-ones row. Note that $D(f) = n$, as there is no good deterministic algorithm to check for such a row. However, $C(f) = \sqrt{n}$, because a 0 in each row, or an all-ones row, are certificates.

Theorem 5 *(Folklore) $D(f) \leq C(f)^2$ for all f. Observe that this is sort of like P versus NP for query complexity, showing that this world is very different than the conjectured world of computational complexity.*

Proof: Pick a 1-certificate A_1 and query it. If $f(X)$ is forced to 1, then halt. Pick another 1-certificate A_2 that is consistent with the first and query it. (If we are ever blocked because the next certificate does not exist, then we halt and output zero.) Iterate in the obvious way.

Each iteration makes $\leq C(f)$ queries. We will prove a bound on the number of iterations in the following lemma. □

Lemma 6 *Every 0-certificate intersects every 1-certificate in at least one place. If you had disjoint certificates, then you simultaneously prove true and false by fixing both of them.*

So, every 1-certificate fixes another bit of a zero-certificate. So our zero certificates shrink by at least 1 each time, so the number of iterations is at most $C(f)$.

6.845 Quantum Complexity Theory
Fall 2010