CMSC 858L: Quantum Complexity

Instructor: Daniel Gottesman

Spring 2023

12 More Bounds From Polynomials, Adversary Method

12.1 References

Lower bounds using polynomials were developed by Beals, Buhrman, Cleve, Mosca, and de Wolf, "Quantum Lower Bounds by Polynomials," quant-ph/9802049. The adversary method was introduced by Ambainis, "Quantum lower bounds by quantum arguments," quant-ph/0002066.

12.2 Examples of polynomial bounds

Let us look at a few example functions and how to bound their query complexity with polynomials. The first example is the unstructured search problem, where you are supposed to compute the OR of all the bits in the bit string to determine if there is a marked element or not. The OR function is the polynomial $p(X_0, \ldots, X_{N-1}) = 1 - \prod_{i=0}^{N-1}(1-X_i)$. This is a degree N polynomial. However, note that Grover's algorithm tells us that there is a degree $O(\sqrt{N})$ polynomial that approximates p. But is this the best we can do?

Another example is the PARITY function, which is 0 if there are an even number of 1s in the input and 1 if there are an odd number of 1s. That is, given an oracle O, are there an even or odd number of inputs that give output 1? Let us figure out a polynomial for PARITY: Let p_n be the polynomial for n variables. Then

$$p_n = X_{n-1}(1 - p_{n-1}) + (1 - X_{n-1})p_{n-1} = X_{n-1} + (1 - 2X_{n-1})p_{n-1}. \tag{1}$$

We get

$$p_n = \sum_{i=0}^{n-1} X_i \prod_{j=i+1}^{n-1} (1 - 2X_j) = \sum_{s=1}^{n} \sum_{i_0 < i_1 < \ldots < i_s} (-2)^{s-1} \prod_{j=1}^{s} X_{i_j}. \tag{2}$$

With N variables, this is a degree N polynomial. But is there a lower degree polynomial that provides a good approximation?

It is not necessarily straightforward to find the approximate degree. Luckily, a lot is known about polynomials and approximating functions by polynomials, since a similar technique also works to set bounds on classical query complexities. One important strategy is to simplify to polynomials of a single variable. One way to do this is to symmetrize by averaging over permutations of the input bits. This gives us a polynomial of the same (or lower) degree that is a function only of the Hamming weight (number of 1s) of the input, since any input with a given Hamming weight can be permutated to any other input of the same Hamming weight, so those two inputs have the same value for the symmetrized polynomial.

If the function f is already symmetric, the polynomial can be directly written in terms of the Hamming weight. In this case, let f_k be the $f(X_0, \ldots, X_{n-1})$ for any $(X_0, \ldots, X_{N-1})$ of Hamming weight k (all give the same value for f_k), and let

$$\Gamma(f) = \min\{|2k - N + 1| \text{ such that } f_k \neq f_{k+1}\}. \tag{3}$$

The classical study of these polynomials produced the following result:

Theorem 1. *If f is a non-constant symmetric Boolean function on $\{0,1\}^N$, then* $\widetilde{\deg}(f) = \Theta(\sqrt{N(N-\Gamma(f))})$.

We can immediately apply this to give a lower bound on unstructured search: The function f is OR, and is symmetric, so the theorem applies. f is 0 for the all-0 input and 1 otherwise; that is, $f_0 = 0$, $f_i = 1$ for $i > 0$. Thus, $\Gamma(f) = N - 1$ and the theorem implies $\widetilde{\deg}(f) = \Theta(\sqrt{N})$. Theorem **??** then implies that the quantum query complexity is $\Omega(\sqrt{N})$.

The method of polynomials is quite powerful. As another example: the PARITY function, which is also symmetric. It has $f_k = 0$ for even k and $f_k = 1$ for odd k. Thus adjacent f_k are always unequal, and the minimum value of $|2k - N + 1|$ is when $k = \lfloor N/2 \rfloor$, and $\Gamma(f)$ is 0 or 1, depending on if N is odd or even. In any case, the theorem then implies that $\widetilde{\deg}(f) = \Theta(N)$ and so we have $Q(\text{PARITY}) = \Theta(N)$ (since N is also an upper bound. This is thus a problem for which quantum computing offers no speedup at all (or rather at most a constant factor speedup; in fact, it offers a factor of 2 speedup). The polynomial method can also be applied to problems with promises or which are not symmetric, but it requires more work and more results about properties of polynomials.

Another application is to limit the possible quantum speedup. The relationship between classical query complexity and approximate degree is not as straightforward as the relationship with quantum query complexity. However, using some more careful analysis of the structure of polynomials, one can show the following:

Theorem 2. *For any total function f (i.e., one with no promise on the input oracle), $D(f) = O(Q(f)^6)$.*

Thus, without a promise, any decision problem can be sped up by at most the 6th power by a quantum algorithm. This is not believed to be tight (although speedups better than quadratic between $Q(f)$ and $R(f)$ are known).

Note that the polynomial method doesn't always give a tight bound. There are problems for which there is a lower degree approximate polynomial than the actual query complexity.

12.3 Adversary method

Another method to bound query complexity is known as the *adversary method* or *quantum adversary method.* In this technique, along with the quantum computer running the query algorithm, there is a second quantum register which governs which oracle we have. This quantum register is in a state which represents a superposition of all possible oracles. In the initial state, the quantum computer has no information about the oracle and so the two registers start out in a tensor product. The oracle calls create entanglement between the two registers, representing the quantum computer learning some information about the oracle. In a successful algorithm, the final state must have some degree of entanglement, and by bounding the amount of entanglement generated by each oracle call, we can set a lower bound on the number of oracle calls needed.

Let us apply this technique to unstructured search. We will restrict attention to oracles that have exactly one marked element; a lower bound on the number of queries needed to solve this problem will also provide a lower bound on the harder problem where the number of marked elements is unknown. In the initial state, the computer is in some standard state, such as $|\psi_0\rangle = |00\ldots0\rangle \otimes \sum_i |O_i\rangle$, where $|O_i\rangle$ indicates the oracle with element i marked. In particular, the second ("oracle") register is in a pure state $\rho_0 = \sum_{ij} \rho_{0,ij} |i\rangle\langle j|$, $\rho_{ij} = 1/N$ when normalized properly.

The final state if the algorithm succeeds with 100% chance is $|\psi_{T-1}\rangle = \sum_i |i\rangle|\alpha_i\rangle|O_i\rangle$ (with some scratch space left in the state $|\alpha_i\rangle$, since the computer has learned the value of the marked element in the oracle. The state is now maximally entangled and the density matrix of the oracle register is now the maximally mixed state: $\rho_{T-1} = \sum_{ij} \rho_{T-1,ij} |i\rangle\langle j|$, $\rho_{T-1,ij} = \delta_{ij}/N$.

If there is not 100% chance of success, the final state can be written as

$$|\psi_{T-1}\rangle = \frac{1}{\sqrt{N}} \sum_i |\xi_i\rangle \otimes |O_i\rangle \tag{4}$$

with proper normalization. (The algorithm doesn't change the oracle register except for a phase during an oracle call, so the absolute value of the amplitude of each oracle option must remain $1/\sqrt{N}$.) When the algorithm succeeds with probability $1-\epsilon > 1/2$, then for all i,

$$|\langle i|\xi_i\rangle| \geq \sqrt{1-\epsilon}. \tag{5}$$

We will need some information about the final density matrix of the oracle register, so let us compute it:

$$\rho_{T-1} = \frac{1}{N}\sum_i |O_i\rangle\langle O_i| + \frac{1}{N}\sum_{i\neq j}\langle\xi_j|\xi_i\rangle|O_i\rangle\langle O_j|. \tag{6}$$

We are particularly interested in the off-diagonal terms, so let us focus on $\rho_{T-1,ij} = \frac{1}{N}\langle\xi_j|\xi_i\rangle$. Let

$$|\xi_i\rangle = a|i\rangle|\omega_{ii}\rangle + b|j\rangle|\omega_{ij}\rangle + c|\eta\rangle \tag{7}$$

$$|\xi_j\rangle = a'|j\rangle|\omega_{jj}\rangle + b'|i\rangle|\omega_{ji}\rangle + c'|\eta'\rangle \tag{8}$$

with $|a|^2 \geq 1-\epsilon$, $|a'|^2 \geq 1-\epsilon$, $|b|^2+|c|^2 \leq \epsilon$, $|b'|^2+|c'|^2 \leq \epsilon$, and $|\eta\rangle$ and $|\eta'\rangle$ both orthogonal to both $|i\rangle$ and $|j\rangle$. Then

$$\langle\xi_j|\xi_i\rangle = ab'\langle\omega_{ji}|\omega_{ii}\rangle + a'b\langle\omega jj|\omega_{ij}\rangle + cc'\langle\eta'|\eta\rangle \tag{9}$$

$$|\langle\xi_j|\xi_i\rangle| \leq |ab'| + |a'b| + |cc'| \tag{10}$$

$$\tag{11}$$

If we fix a, a', b, and c, we can maximize $|ab'|+|a'b|+|cc'|$ by letting the vector (b',c') be parallel to (a,c), so $c'/b' = c/a$. Then if $|b'|^2+|c'|^2 = \epsilon_j$, $|b'|^2(1+c^2/a^2) = \epsilon_2$ and $|a'| = \sqrt{1-\epsilon_2}$, so

$$\begin{aligned}|ab'|+|a'b|+|cc'| &= |ab'|+|a'b|+|c^2b'/a| & (12)\\ &= (|a|+|c^2/a|)\sqrt{\epsilon_j}|a|/\sqrt{|a|^2+|c|^2} + |b|\sqrt{1-\epsilon_j} & (13)\\ &= \sqrt{|a|^2+|c|^2}\sqrt{\epsilon_j} + |b|\sqrt{1-\epsilon_j} & (14)\\ &= |b|(\sqrt{\epsilon_j}+\sqrt{1-\epsilon_j}). & (15)\end{aligned}$$

This is clearly maximized by letting $|b|$ be its maximum value $\sqrt{\epsilon}$. This means the maximum occurs when $c=0$, which also means $c'=0$. Thus,

$$\langle\xi_j|\xi_i\rangle \leq 2\sqrt{\epsilon(1-\epsilon)}. \tag{16}$$

We next must compute how much the entanglement between the two registers can change with a single oracle call. Since the off-diagonal terms start initially at $1/N$ and end at $O(\sqrt{\epsilon(1-\epsilon)}/N)$, we will how much those off-diagonal terms decrease with each call and use this to quantify the entanglement at each stage. In particular, let

$$S = \sum_{i\neq j}|\rho_{ij}|, \tag{17}$$

the sum of the absolute values of the $N(N-1)$ off-diagonal terms. For the initial state, $S_0 = N-1$. In the final state,

$$S_{T-1} \leq \frac{2N(N-1)}{N}\sqrt{\epsilon(1-\epsilon)} = 2(N-1)\sqrt{\epsilon(1-\epsilon)}. \tag{18}$$

Since $\epsilon < 1/2$, $2\sqrt{\epsilon(1-\epsilon)} < 1$ and $S_0 - S_{T-1} = c(N-1)$ for a constant $c = 1-2\sqrt{\epsilon(1-\epsilon)}$.

Let $|\psi^-_{t-1}\rangle = \sum_{ij}\alpha_{ij}|i\rangle|\phi_{ij}\rangle|O_j\rangle$ be the state of the two registers just before the t-th oracle call. Then the state after the t-th oracle call is

$$|\psi^+_{t-1}\rangle = \sum_{i\neq j}\alpha_{ij}|i\rangle|\phi_{ij}\rangle|O_j\rangle - \sum_i \alpha_{ii}|i\rangle|\phi_{ii}\rangle|O_i\rangle. \tag{19}$$

Let us calculate the corresponding density matrices for the oracle register:

$$\begin{aligned}
\rho^-_{t-1} &= \mathrm{Tr}_{\mathrm{computer}} \sum_{ij}\sum_{kk'} \alpha_{ki}\alpha^*_{k'j}|k\rangle|\phi_{ki}\rangle|O_i\rangle\langle k'|\langle\phi_{k'j}|\langle O_j| \qquad (20)\\
&= \sum_{ij}\sum_{k} \alpha_{ki}\alpha^*_{kj}\langle\phi_{kj}|\phi_{ki}\rangle|O_i\rangle\langle O_j| \qquad (21)
\end{aligned}$$

$$\begin{aligned}
\rho^+_{t-1} &= \mathrm{Tr}_{\mathrm{computer}} \sum_{i\neq j}\left[\sum_{k\neq i,k'\neq j} \alpha_{ki}\alpha^*_{k'j}|k\rangle|\phi_{ki}\rangle\langle k'|\langle\phi_{k'j}| - \sum_{k\neq i}\alpha_{ki}\alpha^*_{jj}|k\rangle|\phi_{ki}\rangle\langle j|\langle\phi_{jj}|\right.\\
&\left. - \sum_{k'\neq j}\alpha_{ii}\alpha^*_{k'j}|i\rangle|\phi_{ii}\rangle\langle k'|\langle\phi_{k'j}| + \alpha_{ii}\alpha^*_{jj}|i\rangle|\phi_{ii}\rangle\langle j|\langle\phi_{jj}|\right]|O_i\rangle\langle O_j| + \sum_i \rho_{ii}|O_i\rangle\langle O_i| \qquad (22)\\
&= \sum_{i\neq j}\left[\sum_{k\neq i,j}\alpha_{ki}\alpha^*_{kj}\langle\phi_{kj}|\phi_{ki}\rangle - \alpha_{ji}\alpha^*_{jj}\langle\phi_{jj}|\phi_{ji}\rangle - \alpha_{ii}\alpha^*_{ij}\langle\phi_{ij}|\phi_{ii}\rangle\right]|O_i\rangle\langle O_j| + \sum_i \rho_{ii}|O_i\rangle\langle O_i|. \qquad (23)
\end{aligned}$$

Since our quantification of entanglement doesn't involve the diagonal terms, we don't bother to calculate them.

Then the change in S across the oracle step is

$$\begin{aligned}
|S^+ - S^-| &\leq \sum_{i\neq j}|\rho^+_{t-1,ij} - \rho^-_{t-1,ij}| \qquad (24)\\
&= 2\sum_{i\neq j}|\alpha_{ji}\alpha^*_{jj}\langle\phi_{jj}|\phi_{ji}\rangle + \alpha_{ii}\alpha^*_{ij}\langle\phi_{ij}|\phi_{ii}\rangle| \qquad (25)\\
&\leq 4\sum_{i\neq j}|\alpha_{ji}||\alpha_{jj}| \qquad (26)\\
&= 4\sum_j|\alpha_{jj}|(\sum_{i\neq j}|\alpha_{ji}|). \qquad (27)
\end{aligned}$$

Let $\gamma_j = \sum_i |\alpha_{ji}|^2$. Then, by the Cauchy-Schwartz inequality,

$$\sum_{i\neq j}|\alpha_{ji}| \leq \sqrt{\sum_{i\neq j}1}\sqrt{\sum_{i\neq j}|\alpha_{ji}|^2} = \sqrt{N-1}\sqrt{\gamma_j - |\alpha_{jj}|^2}. \qquad (28)$$

Thus,

$$\begin{aligned}
|S^+ - S^-| &\leq 4\sqrt{N-1}\sum_j|\alpha_{jj}|\sqrt{\gamma_j - |\alpha_{jj}|^2} \qquad (29)\\
&\leq 4\sqrt{N-1}\sqrt{\sum_j|\alpha_{jj}|^2}\sqrt{\sum_j(\gamma_j - |\alpha_{jj}|^2)} \qquad (30)\\
&\leq 4\sqrt{N-1}\sqrt{A(1-A)} \qquad (31)\\
&\leq 2\sqrt{N-1}. \qquad (32)
\end{aligned}$$

again using Cauchy-Schwarz to get the second line and using the normalization condition $\sum_j \gamma_j = 1$ and letting $A = \sum_j |\alpha_{jj}|^2$ to get the third line. Each oracle call drops S by $O(\sqrt{N})$.

The oracle algorithm consists of alternating unitaries on the computer and oracle calls. The unitaries don't affect the oracle density matrix at all and thus don't change S. Therefore, to get a change of $c(N-1)$ in S between the initial and final states, we need $c(N-1)/(2\sqrt{N-1} = O(\sqrt{N})$ oracle calls.

The adversary method as presented here is sometimes stronger than the polynomial method and sometimes weaker. That is, there are some cases where the polynomial method can prove a good lower bound but the adversary method can't and vice-versa. However, the adversary method can be considerably generalized and in principle can give fairly tight bounds in most cases. In practice, though, it may be more tractable to prove things using the polynomial method.