Quantum Computation **(CMU 18-859BB, Fall 2015)**

Lecture 12: Lower Bounds for Element-Distinctness and Collision

October 19, 2015

Lecturer: John Wright *Scribe: Titouan Rigoudy*

1 Outline

In this lecture, we will:

- Briefly review the Element Distinctness problem.
- Introduce the Collision problem and its applications.
- Show how it reduces to the Element Distinctness problem.
- Prove an $O(N^{1/3})$ upper bound on the quantum query complexity of the Collision problem.
- Use the polynomial method to prove a matching $\Omega(N^{1/3})$ lower bound.

2 Review

2.1 Notation

Last lecture we introduced the polynomial method and used it to prove a quantum query lower bound of $\Omega(\sqrt{N})$ for Grover's total search problem.

Recall that in order to do so, we changed the way we view inputs to our quantum algorithms. Instead of considering that inputs are functions $f : [N] \to [M]$, we now consider them to be strings $w \in [M]^N$, with slots labeled by numbers $i \in [N] : w_i \in [M]$. Pictorially:

$$w : \quad \boxed{w_1 \mid w_2 \mid \quad \dots \quad \mid w_N}$$

Correspondingly, our oracle for w is now $\mathcal{O}_w$ instead of $\mathcal{O}_f$, and we care about some properties of strings represented by $F : [M]^N \to \{0, 1\}$.

2.2 Element Distinctness

One such property which we mentioned last time is Element Distinctness (hereafter abbreviated ED):

Definition 2.1. Given $w \in [M]^N$, say "yes" if all the values w_i are distinct, "no" if there is a least one duplicate.

Recall that in the previous lecture we defined two classes of quantum problems: *promise problems*, for which the input w is promised to have some property, i.e. w is restricted to some subset of $[M]^N$; and *total problems*, for which the input w may be any string in $[M]^N$. We saw last time that the quantum query complexity of any total problem cannot be exponentially lower than its classical query complexity.

ED is an example of a *total problem*, and as such we cannot hope to find a quantum query algorithm providing a massive speedup over classical query algorithms. In fact, the classical and quantum query complexities are known to be respectively $\Theta(N)$ and $\Theta(N^{\frac{2}{3}})$.

3 The Collision problem

3.1 Definition

The problem is the following:

Definition 3.1. Given $w \in [M]^N$ and $r \geq 2$ such that $r|N$, output:

- "yes" if all the values in w are distinct, i.e. if w is 1-to-1.
- "no" if w is r-to-1.

Example 3.2. *Let $N = 4, M = 4$ and $r = 2$. The following input is a "yes" instance:*

1	2	3	4

Whereas the following input is a "no" instance:

1	2	2	1

This is clearly a promise problem, as w cannot be any arbitrary string: it must be either 1-to-1 or r-to-1. Also, intuitively, this problem seems easier to solve than ED because of all the repeated elements. One might hope that we may come up with a very efficient quantum query algorithm for solving it.

In the case that $r = 2$, then there is an $O(\sqrt{N})$ randomized query algorithm relying on the birthday paradox. Simply pick $O(\sqrt{N})$ elements from w uniformly at random and a collision will happen with high probability if it is a "yes" instance. This algorithm extends to a general r, yielding a complexity of $O\left(\sqrt{\frac{N}{r}}\right)$.

What about the quantum query complexity? We know that for total problems, there can only be a polynomial gap between classical and quantum query complexities. Here, however, we are considering a promise problem, so we may hope to do much faster than randomized. On the other hand, this problem is very closely related to ED, which cannot be solved very fast. So what will it be?

It turns out that the best quantum query algorithm for the collision problem has a query complexity of $\Theta\left(\left(\frac{N}{r}\right)^{1/3}\right)$. We will dedicate the end of the lecture to proving this.

Remark 3.3. Henceforth, for simplicity of presentation, we shall only consider the case $r = 2$. Everything we will do can be easily generalized to an arbitrary r.

3.2 Applications

Before we dive in to the math, let us ask why we are interested in solving this problem. It is a nice little problem, and there is no reason why not, but there are some more important applications of this problem. Consider an alternate universe in which the collision problem is solvable using only $O(\log N)$ queries, and let us examine the consequences.

Recall the Graph Isomorphism Problem, one of the most famous NP-intermediate problems:

Definition 3.4. Given $G = ([n], E)$ and $G' = ([n], E')$ two graphs of n nodes each, determine if they are *isomorphic*, i.e. if there exists a permutation $\pi \in S_n$ such that:

$$(i, j) \in E \iff (\pi(i), \pi(j)) \in E'$$

In our alternate universe, it is possible to solve this problem efficiently. First, number all permutations of $S_n = \{ \pi_1, \ldots, \pi_{n!} \}$, and write:

$$w = \begin{array}{|c|c|c|c|c|c|}\hline \pi_1(G) & \quad\ldots\quad & \pi_{n!}(G) & \pi_1(H) & \quad\ldots\quad & \pi_{n!}(H) \\ \hline \end{array}$$

Conceptually w is thus a huge string of length $2n!$, even though in practice one may not need to write it all out, as long as individual elements may be computed efficiently as needed. The ”colors” in this case are entire graphs.

If the graphs are isomorphic, then there must be collisions in w. Indeed each permutation of G must appear at least once on the ”left” side of w and on the ”right” side.

In particular, if the graphs are automorphism-free then w is exactly 2-to-1. Running our efficient algorithm for the collision problem on w thus solves the graph isomorphism problem in this case, with $O(\log(2n!)) = O(n \log n)$ queries.

If the graphs are not automorphism-free, then it is also possible to reduce to the collision problem, but how to do so isn't immediately obvious.

Another interesting application of the collision problem is *collision-resistant hash functions.* Such functions are basic primitives in cryptography, allowing to build cryptographically secure systems. Informally, a hash function $h : \{ 0, 1 \}^n \to \{ 0, 1 \}^m$ is collision-resistant if it is ”hard” to find $x, y \in \{ 0, 1 \}^n$ such that $h(x) = h(y)$. In practice, cryptographers pick h at random from a family of hash functions which is proven to resist to known cryptographic attacks with high probability.

However, if we extend our alternate universe to have an efficient solution to the *search problem* version of the collision problem, then there can exist no secure hash functions against quantum computers.

Unfortunately, or fortunately for cryptographers, we do not live in such a universe. There is a known polynomial lower bound for the collision problem.

Interestingly enough, there exists an application of the lower bound to black hole physics. There is a known problem in that field known as the firewall paradox. At a very high level, if a certain number of assumptions related to black hole physics are held to be true, then one can derive a paradox. The problem then becomes that of finding which assumptions are false. A recent paper [HH13] has shown that in order to computationally find those assumptions, it is necessary to solve the collision problem on an intractably large input.

4 Query complexity bounds

4.1 Element Distinctness bounds

In section 2.2 we said that the quantum query complexity of ED is $\Theta(N^{2/3})$. Indeed, there exists a quantum walk-based quantum query algorithm for ED with query complexity $O(N^{2/3})$ [Amb07]. This gives us the upper bound.

Fact 4.1. *The lower bound of for ED follows directly from the collision problem lower bound.*

Proof. By contrapositive. Suppose algorithm $\mathcal{A}$ solves ED with $o(N^{2/3})$ quantum queries. Let $w \in [M]^N$ be either a 1-to-1 string or a 2-to-1 string. The reduction once again follows from the birthday paradox.

Pick a random subset $S \subseteq [N]$ of size $O(\sqrt{N})$. If w is 1-to-1 then $w_{|S}$ is also 1-to-1. Otherwise if w is 2-to-1 then with high probability there is at least one duplicate in $w_{|S}$.

Therefore run $\mathcal{A}$ on $w_{|S}$ and we have solved the collision problem for w using $o\left(\left(N^{1/2}\right)^{2/3}\right) = o(N^{1/3})$ queries. □

4.2 Collision problem upper bound

The upper bound for the collision problem is much easier to prove than the lower bound, so let us tackle it first.

Claim 4.2. *There exists a quantum query algorithm for the collision problem using $O(N^{1/3})$ queries.*

Proof. Let $w \in [M]^N$ be either 1-to-1 or 2-to-1. The algorithm proceeds as follows:

1. Pick a random subset $S \subset [N]$ of size $\Theta(N^{1/3})$.
2. Classically query w on all indices in S, and record all colors in $C = \{\, w_i \mid i \in S \,\}$. If there are any duplicates, say "no" and halt.
3. Implement the following oracle over $N + 1$ qubits:

$$\mathcal{O}_w^C : |i\rangle\,|b\rangle \mapsto \begin{cases} |i\rangle\,|b \oplus 1\rangle & \text{if } w_i \in C \\ |i\rangle\,|b\rangle & \text{if } w_i \notin C \end{cases}$$

   It is possible to implement $\mathcal{O}_w^C$ using two queries to $\mathcal{O}_w$.

   Observe that if w is 2-to-1, then there exists $S' \subset [N] \setminus S$ of size $|S|$ such that $\forall\, i \in S'$, $\mathcal{O}_w^C\,|i\rangle\,|0\rangle = |i\rangle\,|1\rangle$. Conversely, if w is 1-to-1 then $\forall\, i \in [N] \setminus S$, $\mathcal{O}_w^C\,|i\rangle\,|0\rangle = |i\rangle\,|0\rangle$.

4. Run Grover's algorithm on $\mathcal{O}_w^C$ and $[N] \setminus S$. We are looking for one of $|S|$ possible "1" slots out of $N - |S|$, therefore the query complexity is:

$$O\left(\sqrt{\frac{N-|S|}{|S|}}\right) = O\left(\sqrt{\frac{N}{N^{1/3}}}\right) = O(N^{1/3})$$

$\square$

This proves an upper bound of $O(N^{1/3})$ for the collision problem. We will now show that this bound is tight by proving a matching $\Omega(N^{1/3})$ lower bound.

5 A lower bound for the collision problem

5.1 Review: the polynomial method and Grover's algorithm

Recall the polynomial method we introduced in the previous lecture. We want to prove a lower bound on the number of queries needed to compute a given property $F : [M]^N \to \{0, 1\}$.

Let $w \in [M]^N$, and $\forall c \in [M], \forall i \in [N]$, define:

$$\widetilde{w_i^c} = \begin{cases} 1 & \text{if } w_i = c \\ 0 & \text{otherwise} \end{cases}$$

If there exists a t-query quantum algorithm $\mathcal{A}$ to compute F, then there must exist some polynomial P of degree $d = 2t$ in the $\widetilde{w_i^c}$'s such that:

$$\mathbf{Pr}[\mathcal{A} \text{ accepts on } w] = P(w)$$

And therefore, P must approximate F, i.e. for all strings $w \in [M]^N$ the following inequality holds:

$$|P(w) - F(w)| \leq \frac{1}{3}$$

Our goal is then to lower bound d, and we have reduced our quantum query complexity problem to one on approximating polynomials.

Last time we saw how to prove a lower bound for Grover's algorithm using the polynomial method, and the proof was fairly straightforward. This time the proof we will do is fairly involved, so in order to warm up we will review last lecture's proof.

Proof. Let F, the property we are interested in computing be the OR function on N qubits. Suppose we have P, a polynomial of degree d in the $\widetilde{w_i^c}$'s which approximates F.

We want to lower bound d, but P is a complicated multivariate polynomial in NM variables, so the first step is symmetrization. Define Q, another polynomial, as the following:

$$\begin{aligned} Q(w) &= \mathop{\mathbf{E}}_{\pi \sim S_N} \left[P(w_{\pi(1)}, \ldots, w_{\pi(N)}) \right] \\ &= \frac{1}{N!} \sum_{\pi \in S_N} P(w_{\pi(1)}, \ldots, w_{\pi(N)}) \end{aligned}$$

Q is of degree at most d and is symmetric, i.e. $\forall\, w \in [M]^N$, $\forall\, \pi \in S_N$:

$$Q(w_{\pi(1)}, \ldots, w_{\pi(N)}) = Q(w_1, \ldots, w_N)$$

Therefore as shown in Homework 4 there exists a univariate polynomial p of degree at most d such that $\forall\, w \in [M]^N$, $Q(w) = p(|w|)$ where $|w|$ is the Hamming weight of w. In fact, the following equality holds $\forall\, k \in [N]$:

$$p(k) = \mathop{\mathbf{E}}_{|w|=k} [P(w)]$$

The next step is to take p and apply approximation theory to it. We know the following:

$$p(0) \leq \frac{1}{3} \qquad \forall\, i \geq 1,\ p(i) \geq \frac{2}{3}$$

In other words, p makes a sharp jump between 0 and 1, and thereafter is relatively constant. Using Markov's other inequality, we can then show that p must have large degree. □

5.2 Proof of the lower bound

The proof of the collision problem lower bound has some history. First came Aaronson [Aar02], who proved a lower bound of $\Omega((\frac{N}{r})^{1/5})$. Then Shi [Shi02] improved the exponent from 1/5 to 1/3, but only for M large enough. Finally, Ambainis [Amb05] and Kutin [Kut05] both independently proved the 1/3 lower bound for all M.

We will follow Kutin's proof, as it is a nice self-contained argument.

Suppose we have a t-query algorithm for the collision problem. Then there exists a polynomial P of degree $d = 2t$ such that:

$$\begin{aligned} P(w) &\geq \frac{2}{3} \quad \text{if } w \text{ is 1-to-1} \\ P(w) &\leq \frac{1}{3} \quad \text{if } w \text{ is 2-to-1} \\ 0 \leq P(w) &\leq 1 \quad \forall\, w \in [M]^N \end{aligned}$$

What is perhaps surprising in this proof is that eventually, we will care about and closely examine the behavior of P on all inputs *except* those inside the promise, i.e. when w is neither 1-to-1 nor 2-to-1.

The first step is the same as in Grover's lower bound proof: symmetrization. This time however, we need to be more careful. Define $W_{t,a,b}$ to be the set of all strings w such that:

- w is a-to-1 on a set of t indices.
- w is b-to-1 on the remaining $n - t$ indices.
- Elements are otherwise distinct, i.e. there is no element in both sets.

Example 5.1. *The following string is in $W_{4,2,3}$ as it is 2-to-1 on 4 indices ($\{1, 2, 3, 4\}$), and the remaining $10 - 4 = 6$ indices are 3-to-1:*

$$w = \boxed{1}\boxed{1}\boxed{2}\boxed{2}\boxed{3}\boxed{3}\boxed{4}\boxed{3}\boxed{4}\boxed{4}$$

It is also in $W_{6,3,2}$, as we can switch a with b and t with $n - t$.

Example 5.2. *The following string is in $W_{6,3,1}$, as it is 3-to-1 on 6 indices ($\{1, 2, 4, 6, 7, 8\}$), and 1-to-1 on the remaining $8 - 6 = 2$ indices:*

$$w = \boxed{2}\boxed{3}\boxed{1}\boxed{2}\boxed{5}\boxed{1}\boxed{2}\boxed{1}$$

Similarly, it is also in $W_{2,1,3}$.

Example 5.3. *For all t, $W_{t,1,1}$ is the set of all 1-to-1 strings.*

Strings in $W_{t,a,b}$ can therefore be seen as "hybrids" between a-to-1 and b-to-1 strings, leaning more towards the former and less towards the latter as t goes from 0 to N.

Note that $W_{t,a,b}$ is well defined if and only if the following hold:

$$0 \le t \le N \quad a | t \quad b | (N - t)$$

Definition 5.4. Such a (t, a, b) triple is called *valid*.

Now define the following function on valid triples:

$$Q(t, a, b) = \mathop{\mathbf{E}}_{w \sim W_{t,a,b}} [P(w)]$$

Theorem 5.5. *Q is a degree d polynomial in (t, a, b).*

Proof. Maybe in a homework. $\square$

We have now symmetrized our polynomial. The second step in the proof is also conceptually the same as for Grover's: apply approximation theory to Q. This time we still have a 3-variable polynomial, but we will deal with it. But first, we need a small fact that will enable us to analyze Q.

Fact 5.6. *Let $q(X)$ be a degree d polynomial in X. Let $a < b$ be integers, and $z \in [a, b]$. If the following inequalities hold:*

$$|q(i)| \le 2 \quad \forall\, i \in \mathbb{N}, a \le i \le b$$
$$|q(z) - q(\lfloor z \rfloor)| \ge \frac{1}{100}$$

Then $d = \Omega\left(\sqrt{(z - a + 1)(b - z + 1)}\right)$. In particular, $d = \Omega\left(\sqrt{b - a}\right)$.

If additionally $z \approx \frac{a+b}{2}$, then $d = \Omega(b - a)$.

Proof. This is a combination of Markov's other inequality and Bernstein's inequality. □

Now we can finally prove the main result:

Theorem 5.7. *$Q(t,a,b)$ has degree $\Omega(N^{1/3})$.*

Proof. First off, here are some facts about Q we will make use of:

- $\forall\, t \in \{\,0,1,2,\ldots,N\,\}$, $W_{t,1,1}$ is the set of 1-to-1 strings, thus $\frac{2}{3} \leq Q(t,1,1) \leq 1$.
- $\forall\, t \in \{\,0,2,4,\ldots,N\,\}$, $W_{t,2,2}$ is the set of 2-to-1 strings, thus $0 \leq Q(t,2,2) \leq \frac{1}{3}$.
- $\forall\,(t,a,b)$ valid, $0 \leq Q(t,a,b) \leq 1$.

We have reduced the problem to only polynomials that obey these three facts. Now we have to derive the lower bound on d.

Let $T = 2\left\lfloor \frac{N}{4} \right\rfloor$. Observe that $T \approx \frac{N}{2}$, and T is even.

Now consider the 2-D function $(a,b) \mapsto Q(T,a,b)$. We know that:

$$Q(T,1,1) \geq \frac{2}{3} \qquad Q(T,2,2) \leq \frac{1}{3}$$

But what about $Q(T,1,2)$? Either $Q(T,1,2)$ is at least $\frac{1}{2}$, or it is at most $\frac{1}{2}$. Suppose $Q(T,1,2) \leq \frac{1}{2}$.

Define $g(x) = Q(T,1,2x+1)$. Observe that:

$$g(0) = Q(T,1,1) \geq \frac{2}{3} \qquad g\left(\frac{1}{2}\right) = Q(T,1,2) \leq \frac{1}{2}$$

Let k be the following:

$$k = \min\{\, i \in \mathbb{N}^\star \mid |g(i)| > 2 \,\}$$

By definition, $\forall\, i < k$, $|g(i)| \leq 2$. We can therefore apply fact 5.6 and prove that $\deg(g) = \Omega(\sqrt{k})$. Furthermore, we know that $\deg(Q) \geq \deg(g)$ so it follows that $\deg(Q) = \Omega(\sqrt{k})$.

Thus if we could prove that $k = \Omega(N^{2/3})$, then we would be done. The problem is that we have no information about k. Maybe it happens to be tiny, in which case this doesn't prove anything. In order to work around this, we will have to leave 2-D et go back to 3-D.

Let h be the function defined as follows:

$$h(y) = Q(N-(2k+1)y, 1, 2k+1)$$

Observe that the following inequality holds:

$$\left| h\left(\frac{N-T}{2k+1}\right) \right| = |Q(T,1,2k+1)| = |g(k)| > 2$$

In addition, note that $\forall\, i \in \{\,0,1,\ldots,\frac{N}{2k+1}\,\}$, $(N-(2k+1)i, 1, 2k+1)$ is a valid triple. Indeed, the conditions are met:

$$0 \leq N-(2k+1)i \leq N \qquad 1|N-(2k+1)i \qquad 2k+1|(2k+1)i$$

Thus $h(i)$ is well defined, and $|h(i)| \leq 1$. Finally, observe that we chose $T \approx \frac{N}{2}$ so that $\frac{N-T}{2k+1}$ lies approximately halfway between 0 and $\frac{N}{2k+1}$. From there, we can apply fact 5.6 and derive that:

$$\deg(h) = \Omega\left(\frac{N}{2k+1}\right) = \Omega\left(\frac{N}{k}\right)$$

Which, by virtue of the fact that $\deg(Q) \geq \deg(h)$, proves that $\deg(Q) = \Omega\left(\frac{N}{k}\right)$.

In the end there are two cases:

- Either $k \geq N^{2/3}$, in which case $\deg(Q) = \Omega(\sqrt{k}) = \Omega(N^{1/3})$, and we are done.
- Or $k \leq N^{2/3}$, in which case $\deg(Q) = \Omega\left(\frac{N}{k}\right) = \Omega(N^{1/3})$ and we are also done.

This completes the proof of the lower bound for the collision problem in the case where $Q(T,1,2) \leq \frac{1}{2}$. A similar argument involving $Q(T,2,2)$ instead of $Q(T,1,1)$ can be made in the case where $Q(T,1,2) \geq \frac{1}{2}$. $\square$

References

[Aar02] Scott Aaronson. Quantum lower bound for the collision problem. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 635–642. ACM, 2002.

[Amb05] Andris Ambainis. Polynomial degree and lower bounds in quantum complexity: Collision and element distinctness with small range. *Theory of Computing*, 1(1):37–46, 2005.

[Amb07] Andris Ambainis. Quantum walk algorithm for element distinctness. *SIAM Journal on Computing*, 37(1):210–239, 2007.

[HH13] Daniel Harlow and Patrick Hayden. Quantum computation vs. firewalls. *Journal of High Energy Physics*, 2013(6):1–56, 2013.

[Kut05] Samuel Kutin. Quantum lower bound for the collision problem with small range. *Theory of Computing*, 1(1):29–36, 2005.

[Shi02] Yaoyun Shi. Quantum lower bounds for the collision and the element distinctness problems. In *Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on*, pages 513–519. IEEE, 2002.