Quantum algorithms (CO 781/CS 867/QIC 823, Winter 2011)
Andrew Childs, University of Waterloo
LECTURE 20: The collision problem

As the final topic of the course, we will discuss the quantum lower bound for the collision problem. This lower bound is a more involved application of the polynomial method than the simple examples we've seen so far.

Problem statement

In the collision problem, we are given a black-box function $f\colon \{1,\dots,n\} \to \Sigma$ that is promised to be either one-to-one or two-to-one. The goal is to determine which is the case. (Of course, we can assume that n is even, since otherwise the function must be one-to-one.)

Note that for this problem to be well-defined, we must have $|\Sigma| \ge n$. For concreteness, we consider the case where $\Sigma = \{1,\dots,n\}$. Since this is a special case of a function with a larger range, this assumption can only make the problem slightly harder. However, we will present an upper bound that does not depend on $|\Sigma|$, and our lower bound will handle the case where $\Sigma = \{1,\dots,n\}$, so it applies in general.

While it may be most natural to think in terms of one-to-one and two-to-one functions, we can easily cast the collision problem in the framework we used to formally define query complexity. The input alphabet is Σ, and an input is an oracle string $x \in S_0 \cup S_1 \subset \Sigma^n$, where

$$S_0 = \{x \in \Sigma^n \colon \forall\, i \ne j,\ x_i \ne x_j\} \tag{1}$$
$$S_1 = \{x \in \Sigma^n \colon \forall\, i \text{ there is a unique } j \text{ such that } x_i = x_j\}. \tag{2}$$

In other words, we are promised that $x \in S_0 \cup S_1$, i.e., that either every character of the string is distinct or that every character of x appears exactly twice. Again, the goal is to determine which of these is the case, so we are computing a function $\textsc{collision}\colon S_0 \cup S_1 \to \{0,1\}$ with $\textsc{collision}(x) = y$ if $x \in S_y$.

Why is the collision problem interesting? We know that the unstructured search problem takes $\Omega(\sqrt{n})$ queries, i.e., exponentially many in the number of bits required to specify an index. Thus quantum computers cannot provide a speedup for completely unstructured problems. The collision problem is a problem with slightly more structure: instead of looking for a needle in a haystack, we are merely looking for two identical items among a stack consisting of two pieces of hay, two needles, two toothpicks, two twigs, two cactus spines, etc. We will see that the quantum query complexity of the collision problem is $\Theta(n^{1/3})$, so while this more structured problem can be solved slightly faster than unstructured search, it still requires many queries. In other words, the kind of structure present in the collision problem is not sufficient to allow an exponential quantum speedup.

Note that the structure in the collision problem arises naturally in the hidden subgroup problem: if the hidden subgroup has order r, then the hiding function is r-to-one. But the HSP has additional structure (namely, that of the underlying group), which explains why its quantum query complexity is so much smaller.

Classical query complexity

Before investigating quantum algorithms and lower bounds, let's briefly consider the classical query complexity of the collision problem. A natural algorithm for finding a collision in a two-to-one function is as follows: query random indices until we observe a collision. For two independently random indices $i, j \in \{1, \ldots, n\}$, we have $\Pr(x_i = x_j) = \frac{1}{n-1}$. If we query m indices in this way, there are $\binom{m}{2}$ pairs, so the expected number of collisions seen is $\binom{m}{2}/(n-1)$. With $m = \sqrt{n}$ this is $\Omega(1)$, so we expect to see a collision. Indeed, a second moment argument shows that this happens with constant probability.

In fact, this algorithm is optimal. Until we see a collision, there is no better strategy than to query randomly. By the union bound, the probability of seeing a collision after making m queries is at most $\binom{m}{2}/(n-1) = O(m^2/n)$, so $m = \Omega(\sqrt{n})$.

Quantum algorithm

Now let's consider quantum algorithms for the collision problem. One simple approach is a trivial application of Grover's algorithm: first query x_1, and then search over $x_2, x_3, \ldots, x_n$ for x_1. This approach uses $O(\sqrt{n})$ queries, so it is no better than the above classical algorithm, although it achieves the same running time in a very different way.

However, we can improve the performance by combining this approach with the main idea of the optimal classical algorithm. The approach is to query m distinct indices (it doesn't matter how we choose them) and check for a collision. If we find a collision then we are done; otherwise, we can search the remaining $n - m$ characters for one of the m values we have seen. Overall, this approach takes time $m + O(\sqrt{(n-m)/m})$. This is minimized by taking $m = O(n^{1/3})$, showing that $Q(\textsc{collision}) = O(n^{1/3})$.

This is similar to an algorithm for element distinctness that we saw previously. The difference is that in element distinctness, we could have a unique collision, so a single run of the basic algorithm only succeeds with small probability; we address this by performing amplitude amplification, at the cost of a longer running time. Recall that the resulting algorithm for element distinctness took time $O(n^{3/4})$, which turned out to be suboptimal. In contrast, the above algorithm for the collision problem turns out to be optimal.

As a side remark, note that this algorithm uses a lot of space, since we need to store the $m = n^{1/3}$ items queried initially. It is an open problem to understand the quantum query complexity of the collision problem with restricted space (i.e., to find a time-space tradeoff).

Toward a quantum lower bound

How can we prove a quantum lower bound for the collision problem?

Note that the positive-weighted adversary method will not be sufficient. This is a consequence of the so-called "property-testing barrier": for a decision problem in which "yes" input strings differ from "no" input strings in an ϵ fraction of their characters, the best possible positive adversary lower bound goes like $1/\epsilon$. For the collision problem, this means that the adversary method can at best prove a constant lower bound. (While we now know that the negative adversary method can in principle prove an $\Omega(n^{1/3})$ lower bound for collision, finding an explicit negative adversary lower bound remains an open question).

Thus our strategy will be to apply the polynomial method. To do this, we need to adapt the method to the case of a non-binary input alphabet. To do this, we define variables δ_{ij} for

$i, j \in \{1, \ldots, n\}$, as follows:

$$\delta_{ij} := \begin{cases} 1 & \text{if } x_i = j \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Then we claim that acceptance probability of a t-query quantum algorithm is a multilinear polynomial in the $\{\delta_{ij}\}$ of degree at most $2t$.

This can be proved along similar lines to the binary case. Suppose we use an addition modulo n query, where $|i, j\rangle \mapsto |i, j + x_i\rangle$. Then we have

$$\begin{aligned} |i, j + x_i\rangle &= \sum_k \delta_{ik} |i, j + k\rangle & (4) \\ &= \sum_\ell \delta_{i,\ell-j} |i, \ell\rangle, & (5) \end{aligned}$$

so the degree of each amplitude can increase by at most 1 when we make a query.

Next we would like to obtain a simpler polynomial. We cannot directly symmetrize over the variables $\{\delta_{ij}\}$, as this would destroy too much of the structure of the problem.

The original idea leading to a nontrivial collision lower bound, due to Aaronson, was to express the acceptance probability as a bivariate polynomial in n and r, where the function is r-to-one. The main difficulty with this approach is that we need to have $r \mid n$ in order for such inputs to make sense (so that we can at least say that the acceptance probability of a quantum algorithm is defined, and hence a given approximating polynomial is bounded between 0 and 1). This approach originally gave a lower bound $\Omega(n^{1/5})$, which can easily be improved to $\Omega(n^{1/4})$. Subsequently, Shi improved this to give the optimal result of $\Omega(n^{1/3})$.

Constructing the functions

In this lecture we will describe a later argument due to Kutin that seems to be simplest known approach to lower bounding the collision problem, yet that gives a tight lower bound, and even covers the case where the range is $\{1, \ldots, n\}$. We will follow Kutin's presentation closely, with only some slight changes of notation.

The basic idea of Kutin's proof is to consider functions that are a-to-one on part of the domain and b-to-one on the rest of the domain. Let us say that a triple (m, a, b) is valid if $m \in \{0, 1, \ldots, n\}$, $a \mid m$, and $b \mid n - m$. Then we define an input $x^{m,a,b}$ as

$$x_i^{m,a,b} = \begin{cases} \lceil i/a \rceil & \text{if } 1 \le i \le m \\ n - \lfloor (n-i)/b \rfloor & \text{if } m < i \le n. \end{cases} \tag{6}$$

Given such an input, we can obtain a family of inputs by permuting the input alphabet and the characters of the string arbitrarily. Specifically, for any permutations $\sigma\tau$ of $\{1, \ldots, n\}$, we define an input $x^{\sigma\tau}$ with $x_i^{\sigma\tau} := \tau(x_{\sigma(i)}^{m,a,b})$. This induces corresponding binary variables $\delta_{ij}^{\sigma\tau}$ with $\delta_{ij}^{\sigma\tau} = 1$ iff $x_i^{\sigma\tau} = j$.

Now we claim that the acceptance probability of a quantum algorithm presented with such an input is a polynomial in m, a, b.

Lemma. *Let $p(\{\delta_{ij}\})$ be a polynomial in the δ_{ij}. For any valid triple (m, a, b), let*

$$P(m, a, b) := \mathop{\mathbb{E}}_{\sigma,\tau} [p(\{\delta_{ij}^{\sigma\tau}\})]. \tag{7}$$

Then P is a polynomial in m, a, b with $\deg(P) \le \deg(p)$.

Proof. Any polynomial p can be expanded as a sum of monomials of the form

$$I_S(\{\delta_{ij}\}) := \prod_{j=1}^{n} \prod_{i \in S_j} \delta_{ij} \tag{8}$$

where $S_1, \ldots, S_n$ are disjoint subsets of $\{1, \ldots, n\}$. This monomial is 1 if all the indices $i \in S_j$ satisfy $x_i = j$, and 0 otherwise. Its degree is $s := \sum_j |S_j|$. It suffices to handle the case where p is any particular monomial.

For any input $x^{\sigma\tau}$, we can identify the subset of function values in the range of the a-to-one part of the function and the b-to-one part of the function. Given $U \subseteq \{1, \ldots, n\}$, let X denote the event that all function values in U belong to the a-to-one part and all function values not in U belong to the b-to-one part (more formally, that $\tau(j) \in U$ for all $j \in \{1, \ldots, \frac{m}{a}\}$ with $S_j \ne \emptyset$ and $\tau(j) \notin U$ for all $j \in \{n, n-1, \ldots, n - \frac{n-m}{b} + 1\}$ with $S_j \ne \emptyset$). We collect together the inputs corresponding to a particular event X. Then we have

$$\begin{aligned} P(m, a, b) &= \sum_{U \subseteq \{1,\ldots,n\}} \Pr_\tau[X] \mathop{\mathbb{E}}_{\sigma,\tau}[I_S(\{\delta_{ij}^{\sigma\tau}\}) \mid X] & (9) \\ &= \sum_{U \subseteq \{1,\ldots,n\}} \Pr_\tau[X] \Pr_{\sigma,\tau}[\forall\, j,\ \forall\, i \in S_j,\ \delta_{ij}^{\sigma\tau} = 1 \mid X]. & (10) \end{aligned}$$

Let $u := |U|$, and let t denote the number of values of j with $|S_j| \ne 0$. Then we have

$$\begin{aligned} \Pr_\tau[X] &= \frac{\frac{m}{a}(\frac{m}{a} - 1) \cdots (\frac{m}{a} - u + 1) \cdot (\frac{n-m}{b})(\frac{n-m}{b} - 1) \cdots (\frac{n-m}{b} - (t-u) + 1) \cdot (n-t)!}{n!} & (11) \\ &= \frac{(\frac{m}{a})_u (\frac{n-m}{b})_{t-u}}{(n)_t} & (12) \end{aligned}$$

where

$$(n)_k := \frac{n!}{(n-k)!} = n(n-1) \cdots (n-k+1). \tag{13}$$

Here the numerator of (11) has three contributions: the number of ways to permute the u function values in the a-to-one part is $(\frac{m}{a})_u$, the number of ways to permute the $t-u$ function values in the b-to-one part is $(\frac{n-m}{b})_{t-u}$, and the number of ways to permute the $n-t$ remaining function values is $(n-t)!$. Note that this expression is a rational function in m, a, b whose numerator has degree t and whose denominator is $a^u b^{t-u}$.

Now consider the latter term of (10). Given that X occurs, $\Pr_\tau[\forall\, j,\ \forall\, i \in S_j,\ \delta_{ij}^{\sigma\tau} = 1]$ is independent of σ, so suppose σ is any permutation such that X occurs. In other words, we only need to count consistent ways of permuting the indices i. Observe that the number of ways to permute the indices i such that $x_i = j$ for some $j \in U$ is

$$a(a-1) \cdots (a - |S_j| + 1) = (a)_{|S_j|}. \tag{14}$$

Similarly, the number of ways to permute the indices i such that $x_i = j$ for some $j \notin U$ is $(b)_{|S_j|}$. In addition, we can permute the remaining $n-s$ indices however we like. Thus we have

$$\begin{aligned} \Pr_{\sigma,\tau}[\forall\, j,\ \forall\, i \in S_j,\ \delta_{ij}^{\sigma\tau} = 1 \mid X] &= \frac{(n-s)! \prod_{j \in U} (a)_{|S_j} \prod_{j \notin U} (b)_{|S_j|}}{n!} & (15) \\ &= \frac{\prod_{j \in U} (a)_{|S_j|} \prod_{j \notin U} (b)_{|S_j|}}{(n)_s}. & (16) \end{aligned}$$

This expression is a polynomial in a, b of degree s. Also, it is divisible by a^u and b^{t-u}. Thus $P(m, a, b)$ is a polynomial in m, a, b of degree $t + s - u - (t - u) = s$. $\square$

Finishing the proof

To prove the lower bound, we will use the following fact about real polynomials. (Note that this fact is also useful elsewhere, e.g., in proving a tight lower bound for general threshold functions.)

Lemma (Paturi)**.** *Let $f \in \mathbb{R}[x]$, let $a, b \in \mathbb{Z}$ with $a < b$, and let $\xi \in [a, b]$. If there are constants c, d such that*

- *$|f(i)| \le c$ for all integers $i \in [a, b]$*
- *$|f(\lfloor \xi \rfloor) - f(\xi)| \ge d$,*

then $\deg(f) = \Omega(\sqrt{(\xi - a + 1)(b - \xi + 1)})$.

Regardless of ξ, Paturi's lemma always shows that the degree is $\Omega(\sqrt{b-a})$. If ξ is near the middle of the range then it does much better, showing that the degree is $\Omega(b - a)$. Also note that by continuity, it is sufficient for f to differ by a constant amount between two consecutive integers.

Now we are ready to prove that the quantum query complexity of the collision problem is $\Omega(n^{1/3})$. Let $p(\{\delta_{ij}\})$ be the acceptance probability of a t-query quantum algorithm that solves the collision problem with error probability at most $1/3$, and let $P(m, a, b)$ be as above. We know that $t \ge \deg(P)/2$. Furthermore, we know that P has the following properties:

- $0 \le P(m, 1, 1) \le 1/3$
- $2/3 \le P(m, 2, 2) \le 1$
- $0 \le P(m, a, b) \le 1$ for any valid triple (m, a, b).

Now consider inputs that are roughly half one-to-one and half two-to-one. For concreteness, let $m = 2\lfloor n/4 \rfloor$. Since $n - m$ is even (recall that n is always even by assumption since otherwise the problem is trivial), $(m, 1, 2)$ is valid. We consider two cases, depending on whether the algorithm is more likely to call this a yes or a no input. First suppose $P(m, 1, 2) \ge 1/2$.

Let r be the smallest integer such that $|P(m, 1, r)| \ge 2$. First we consider $P(m, 1, x)$ as a function of x. For all $x \in \{1, \ldots, r - 1\}$, we have $-2 \le P(m, 1, r) \le 2$. But we also know that $|P(m, 1, 1) - P(m, 1, 2)| \ge \frac{1}{2} - \frac{1}{3} = \frac{1}{6}$. By Paturi's lemma, this implies that $\deg(P) = \Omega(\sqrt{r})$.

On the other hand, consider the polynomial $g(x) := P(n - rx, 1, r)$. When $x \in \mathbb{Z}$ such that $rx \in \{0, \ldots, n\}$, the triple $(n - rx, 1, r)$ is valid, so we have $0 \le g(x) \le 1$ for all integers $x \in [0, \lfloor n/r \rfloor]$. However, $|g(m/r)| = |P(n, 1, r)| \ge 2$, and $\lfloor m/r \rfloor$ is about halfway between 0 and $\lfloor n/r \rfloor$. Thus by Paturi's lemma, $\deg(P) = \Omega(n/r)$.

Combining these results, we have $\deg(P) = \Omega(\sqrt{r} + n/r)$. The weakest lower bound is obtained when the two terms are equal, i.e., when $r = \Theta(n^{2/3})$; therefore $\deg(P) = \Omega(n^{1/3})$.

It remains to consider the case where $|P(m, 1, r)| < 1/2$. But the same conclusion holds here by a very similar argument. (Let r be the smallest *even* integer for which $|P(m, r, 2)| \ge 2$; on the one hand, $\deg(P) = \Omega(\sqrt{d})$ as before, but on the other hand, the polynomial $h(x) := P(rx, r, 2)$ shows that $\deg(P) = \Omega(n/r)$.)

Overall, it follows that any quantum algorithm for solving the collision problem must use $\Omega(n^{1/3})$ queries.