COL7160 : Quantum Computing

Lecture 10: Lower Bound for Simon's Problem

Instructor: Rajendra Kumar **Scribe:** Vijay Balaji

1 Recap: Simon's Problem

In the last class, we discussed Simon's problem, which is one of the earliest examples demonstrating exponential quantum speedup over classical algorithms.

1.1 Problem Statement

Definition 1 (Simon's Problem)**.** We are given a function $f : \{0,1\}^n \to \{0,1\}^n$ with the following promise:
Promise: There exists an unknown string $s \in \{0,1\}^n$ such that for all $x, y \in \{0,1\}^n$,

$$f(x) = f(y) \iff y = x \text{ or } y = x \oplus s,$$

where $\oplus$ denotes bitwise XOR.
Goal: Find the hidden string s.

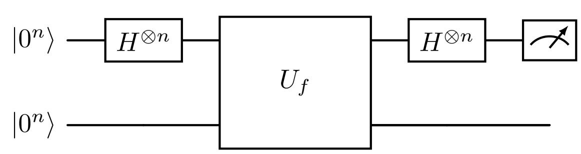
In other words, the function f is either one-to-one (when $s = 0^n$) or exactly two-to-one (when $s \neq 0^n$), where each output has exactly two pre-images x and $x \oplus s$.

1.2 Simon's Quantum Algorithm

The quantum algorithm for Simon's problem with the measurement at the end proceeds as follows. Remember that we were given access to an oracle U_f that implements:

$$U_f : |x\rangle |b\rangle \mapsto |x\rangle |b \oplus f(x)\rangle .$$

The quantum circuit for Simon's algorithm is:



Let us trace through the state evolution:
Step 1: Initialize the first register to $|0^n\rangle$ and second register to $|0^n\rangle$:

$$|\psi_0\rangle = |0^n\rangle |0^n\rangle .$$

Step 2: Apply Hadamard gates to the first register:

$$|\psi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle |0^n\rangle .$$

Step 3: Apply the oracle U_f:

$$|\psi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle |f(x)\rangle .$$

Since $f(x) = f(x \oplus s)$ for all x, we can rewrite this as:

$$|\psi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{z \in \text{Image}(f)} (|x_z\rangle + |x_z \oplus s\rangle) |z\rangle ,$$

where x_z is some fixed pre-image of z.
Step 4: Apply Hadamard gates to the first register. For any $x \in \{0,1\}^n$, we have:

$$H^{\otimes n} |x\rangle = \frac{1}{\sqrt{2^n}} \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y} |y\rangle ,$$

where $x \cdot y = \sum_{i=1}^{n} x_i y_i \pmod 2$ is the inner product modulo 2.
Applying this to our state:

$$\begin{aligned} |\psi_3\rangle &= \frac{1}{2^n} \sum_{z \in \text{Image}(f)} \sum_{y \in \{0,1\}^n} \left[(-1)^{x_z \cdot y} + (-1)^{(x_z \oplus s) \cdot y} \right] |y\rangle |z\rangle \\ &= \frac{1}{2^n} \sum_{z \in \text{Image}(f)} \sum_{y \in \{0,1\}^n} (-1)^{x_z \cdot y} \left[1 + (-1)^{s \cdot y} \right] |y\rangle |z\rangle . \end{aligned}$$

The key observation is that the term $[1 + (-1)^{s \cdot y}]$ equals:

$$1 + (-1)^{s \cdot y} = \begin{cases} 2 & \text{if } s \cdot y = 0 \pmod 2, \\ 0 & \text{if } s \cdot y = 1 \pmod 2. \end{cases}$$

Therefore, when we measure the first register, we obtain a uniformly random $y \in \{0,1\}^n$ such that $s \cdot y = 0 \pmod 2$. This gives us one linear equation over $\mathbb{F}_2$ (the field with two elements) satisfied by s.
Step 5: Repeat the algorithm $O(n)$ times to collect $n - 1$ linearly independent equations of the form $s \cdot y_i = 0$. By solving these linear equations we get a unique non-zero solution $s \in \{0,1\}^n$. If $f(0) = f(s)$ then it is a two-to-one function with the promise that $f(x) = f(x \oplus s)$. Otherwise, f is a one-to-one function.

1.3 Analysis

The quantum algorithm requires $O(n)$ queries to the oracle and $O(n^3)$ classical post-processing time to solve the system of linear equations. This provides an exponential speedup over any classical algorithm, as we will see in the next section.

Note 2. The principle demonstrated here is general: it is always possible to defer all measurements to the end of a quantum circuit. Refer to the exercise 7 of chapter 2 in Ronald's Notes [dW23].

2 Classical Algorithms for Simon's Problem

2.1 Classical Randomized Algorithm

A natural randomized approach is to query the function at random inputs and check for collisions.

Algorithm 1 Randomized Algorithm for Simon's Problem

Input: Oracle access to $f : \{0,1\}^n \to \{0,1\}^n$ satisfying Simon's promise
Output: The hidden string s
Choose $x_1, x_2, \ldots, x_T$ uniformly at random from $\{0,1\}^n$
for $i = 1$ to T **do**
    for $j = 1$ to $i - 1$ **do**
        if $f(x_i) = f(x_j)$ **then**
            return $s = x_i \oplus x_j$
        end if
    end for
end for
return $s = 0^n$ ▷ No collision found

Theorem 3. *Let $X = 2^{n-1}$ be the number of pairs in the partition induced by s. If we make T random queries, the probability of not finding a collision is at most $e^{-T^2/(2X)} = e^{-T^2/2^n}$.*

Proof. For each query x_i, let A_i be the set of all previously queried outputs. Initially, $|A_1| = 0$. After $i-1$ queries, $|A_i| = i - 1$.
The probability that $f(x_i)$ does not collide with any element in A_i is:

$$\Pr[\text{no collision at step } i \mid \text{no collision in steps } 1, \dots, i-1] = 1 - \frac{|A_i|}{X}.$$

This is because if the previous $i-1$ queries sampled from distinct pairs, then there are $X - (i-1)$ pairs left, but we need the new query to hit one of the $i-1$ already-sampled pairs to get a collision. The probability of this is $\frac{i-1}{X}$.
The probability of no collision after T queries is:

$$\begin{aligned}\Pr[\text{no collision}] &= \prod_{i=1}^{T} \left(1 - \frac{i-1}{X}\right) \\ &= \prod_{i=1}^{T} \left(1 - \frac{i-1}{2^{n-1}}\right).\end{aligned}$$

Using the inequality $1 - x \leq e^{-x}$ for $x \geq 0$:

$$\begin{aligned}\Pr[\text{no collision}] &\leq \prod_{i=1}^{T} \exp\left(-\frac{i-1}{2^{n-1}}\right) \\ &= \exp\left(-\frac{1}{2^{n-1}} \sum_{i=0}^{T-1} i\right) \\ &= \exp\left(-\frac{T(T-1)}{2 \cdot 2^{n-1}}\right) \\ &\leq \exp\left(-\frac{T^2}{2^n}\right).\end{aligned}$$

□

Corollary 4. *To achieve a constant success probability (say, $1/2$), we need $T = O(\sqrt{2^n}) = O(2^{n/2})$ queries.*

Proof. Setting $e^{-T^2/2^n} \leq 1/2$, we get:

$$-\frac{T^2}{2^n} \leq \ln(1/2) = -\ln 2,$$

which gives $T^2 \geq 2^n \ln 2$, so $T = O(2^{n/2})$. □

2.2 Lower Bound for Randomized Algorithms

Claim 5. *No randomized classical algorithm can solve Simon's problem with constant success probability using significantly fewer than $\Theta(2^{n/2})$ queries.*

Intuition: Suppose we have made k queries and obtained values $x_1, x_2, \dots, x_k$ with no collisions. Then s does not belong to the set:

$$S_k = \{x_i \oplus x_j : 1 \leq i < j \leq k\}.$$

The size of S_k is at most $\binom{k}{2} = \frac{k(k-1)}{2}$. The total number of possible non-zero values for s is $2^n - 1$. Therefore, the number of remaining possible values for s is at least:

$$2^n - 1 - \binom{k}{2}.$$

By making one additional query x_{k+1}, we can check whether $s \in \{x_i \oplus x_{k+1} : 1 \leq i \leq k\}$, which has size k. The probability that we find a collision at step $k+1$, given no collision in the first k steps, is:

$$\Pr[\text{collision at step } k+1 \mid \text{no collision in steps } 1, \ldots, k] = \frac{k}{2^n - 1 - \binom{k}{2}}.$$

This probability remains small until k is close to $2^{n/2}$. More formally:

$$\begin{aligned}
\Pr[\text{no collision after } k \text{ queries}] &= \prod_{i=1}^{k} \left(1 - \frac{i-1}{2^n - 1 - \binom{i-1}{2}}\right) \\
&= \prod_{i=1}^{k} \left(\frac{2^n - 1 - \binom{i-1}{2} - i + 1}{2^n - 1 - binomi - 12}\right) \\
&= \prod_{1=1}^{k} \left(\frac{2^n - 1 - \binom{i}{2}}{2^n - 1 - \binom{i-1}{2}}\right) \\
&\geq 1 - \frac{k^2}{2^n},
\end{aligned}$$

For constant success probability, we need $k^2/2^n = \Theta(1)$, which gives $k = \Theta(2^{n/2})$.
This analysis shows that Simon's problem exhibits an exponential quantum speedup: quantum algorithms require $O(n)$ queries while classical algorithms (both deterministic and randomized) require $\Omega(2^{n/2})$ queries. This was one of the first concrete examples demonstrating the power of quantum computation.
In the following reference, they have proven the lower bound for deterministic algorithms as well similarly [?]

3 Integer Factorization

We now turn to one of the most celebrated applications of quantum computing: Shor's algorithm for integer factorization. This algorithm demonstrates quantum supremacy on a problem of immense practical importance.

3.1 Problem Statement

Definition 6 (Integer Factorization)**.**
Input: A positive integer $N > 2$.
Output: The prime factorization of N, i.e., express N as:

$$N = p_1^{k_1} \cdot p_2^{k_2} \cdot \ldots \cdot p_m^{k_m},$$

where $p_1, p_2, \ldots, p_m$ are distinct primes and $k_1, k_2, \ldots, k_m$ are positive integers.

Integer factorization is believed to be computationally hard for classical computers.

4 Order Finding Problem

Definition 7 (Order Finding Problem)**.**
Input: Positive integers N and $a < N$ satisfying $\gcd(a, N) = 1$.
Output: The smallest positive integer r such that:

$$a^r \equiv 1 \pmod{N}.$$

This integer r is called the *order* of a modulo N.

Remark 8. The order r always exists and divides $\varphi(N)$, where φ is Euler's totient function. By Euler's theorem, $a^{\varphi(N)} \equiv 1 \pmod{N}$ for any a coprime to N.

5 Period Finding Problem

The order finding problem can be reformulated as a period finding problem on a particular function.

Definition 9 (Period Finding Problem)**.**
Given: Oracle access to a function $f : \{0, 1, \ldots, n-1\} \to \{0, 1, \ldots, n-1\}$.
Promise: There exists an integer r (the period) such that for all $x \in \{0, 1, \ldots, n-1\}$:

$$f(x) = f(x + r \bmod n).$$

Output: Find the period r.

For the order finding problem with modulus N and base a, we define:

$$f(x) = a^x \bmod N.$$

Then f is periodic with period r, the order of a modulo N, because:

$$f(x+r) = a^{x+r} = a^x \cdot a^r \equiv a^x \cdot 1 = a^x = f(x) \pmod N.$$

6 Phase Estimation Problem

Definition 10 (Phase Estimation Problem:)**. Given:**

- A unitary operator U (given as a quantum circuit or as an oracle).
- A *quantum* state $|\psi\rangle$ of U such that:

  $$U |\psi\rangle = \lambda |\psi\rangle,$$

  where λ is the corresponding eigenvalue.

Since U is unitary, we must have $|\lambda| = 1$, so we can write:

$$\lambda = e^{2\pi i\theta}$$

for some $\theta \in [0, 1)$.
Goal: Estimate the phase θ to a desired precision.

References

[dW23] Ronald de Wolf. Quantum computing: Lecture notes, 2023.