Quantum Computing (5314QUCO6Y), Final exam

Ronald de Wolf

June 12, 2017
14:00–17:00
IWO 4.04B, Meibergdreef 29, Amsterdam ZO

The exam is "open book," meaning you can bring any kind of paper you want but no electronic devices. Please answer in English. Use a black or blue pen, not a pencil. Write clearly and explicitly, and explain your answers. For a multipart-question, you may assume answers for earlier parts of the question to answer later parts, even if you don't know the earlier answers. The total number of points adds up to 9; your exam grade will be your number of points +1. Your final grade will be 60% exam + 40% homework, rounded to the nearest integer.

1. **(1 point)**
   (a) Which quantum state do we get if we apply $(H \otimes I)$CNOT to
   $$\sqrt{\frac{1}{3}}|00\rangle + \sqrt{\frac{2}{3}}|11\rangle?$$
   Here I is the 1-qubit identity operation, H is the 1-qubit Hadamard, and CNOT is the 2-qubit controlled-not operation with the first (=leftmost) qubit being the control.
   (b) What is the probability of seeing $|11\rangle$ if we measure the resulting state in the computational basis?

2. **(2 points)** The 3-bit majority function $f : \{0,1\}^3 \to \{0,1\}$ takes value 1 iff at least 2 of its 3 input bits are 1.
   (a) Give a quantum algorithm that computes $f(x)$ with success probability 1 (for every possible input $x \in \{0,1\}^3$), using 2 queries. You do not need to give the exact circuit in full detail, an informal (but precise) description of the algorithm is good enough.
   Hint: Remember that we can compute the parity of 2 bits with 1 quantum query.
   (b) Prove a corresponding lower bound: 2 queries are also *necessary* for every quantum algorithm that computes f with success probability 1.
   (c) What is the quantum query complexity of this f if you allow an algorithm to have error probability at most 1/3 on every input?

3. **(2.5 points)** Consider the search problem: we have oracle access to $x \in \{0,1\}^N$, with unknown Hamming weight $t = |x|$. We want to find a solution, i.e., an index $i \in \{0, \ldots, N-1\}$ such that $x_i = 1$. If $x = 0^N$ then our search algorithm should output "no solution."

   (a) Suppose we know an integer s such that $t \in \{1, \ldots, s\}$. Give a quantum algorithm that finds a solution with probability 1, using $O(\sqrt{sN})$ queries to x.

   Hint: Try running the exact version of Grover (see top of p.38 of the lecture notes) with different guesses for what the actual t is.

   (b) Suppose we know that $t \in \{s+1, \ldots, N\}$. Give a quantum algorithm that finds a solution with probability at least $1 - 2^{-s}$, using $O(\sqrt{sN})$ queries to x.

   (c) For given $\varepsilon > 0$, give a quantum algorithm that solves the search problem with probability $\geq 1 - \varepsilon$ using $O(\sqrt{N \log(1/\varepsilon)})$ queries, without assuming anything about t.

   NB: The important part here is that the $\log(1/\varepsilon)$ is inside the square-root; usual amplification by $O(\log(1/\varepsilon))$ repetitions of basic Grover would give the worse upper bound of $O(\sqrt{N}\log(1/\varepsilon))$ queries.

4. **(1.5 points)** Explain how Simon's problem (Chapter 3 of the notes) may be viewed as an instance of the Abelian Hidden Subgroup Problem. Say explicitly what the groups G and H are, what the function f is, and why these satisfy the requirements of the HSP. Also say explicitly what the QFT corresponding to G is, what the group $H^{\perp}$ is, and why sampling from $H^{\perp}$ a small number of times leads to an efficient solution to Simon's problem.

5. **(2 points)** This question is about the classical and quantum communication complexity of the n-bit equality function: Alice gets input $x \in \{0,1\}^n$, Bob gets input $y \in \{0,1\}^n$, and they have to decide whether $x = y$. Alice and Bob do not share randomness (or entanglement) but can use local (private) randomness.

   (a) Fix a prime number $p \in [3n, 6n]$, then the set $\mathbb{F}_p$ of integers modulo p is a finite field (i.e., it has a well-defined addition and multiplication). For $x = (x_0, \ldots, x_{n-1}) \in \{0,1\}^n$, define the univariate polynomial $P_x : \mathbb{F}_p \to \mathbb{F}_p$ of degree $< n$ as $P_x(t) = \sum_{i=0}^{n-1} x_i t^i$ (note that the n bits of x are used as coefficients here, not as the argument of the polynomial). Show that for distinct n-bit strings x and y, we have $\Pr_{t \in \mathbb{F}_p}[P_x(t) = P_y(t)] \leq 1/3$, where the probability is taken over a uniformly random $t \in \mathbb{F}_p$.

   Hint: Two distinct polynomials, each of degree $\leq d$, are equal on at most d points of the domain $\mathbb{F}_p$.

   (b) Use (a) to give a classical communication protocol where Alice sends an $O(\log n)$-bit message to Bob, and Bob can decide whether $x = y$ with success probability $\geq 2/3$.

   (c) Use (a) to give a quantum fingerprinting scheme $x \mapsto |\phi_x\rangle$, where quantum state $|\phi_x\rangle$ has $O(\log n)$ qubits, and $|\langle \phi_x | \phi_y \rangle| \in [0, 1/3]$ for all distinct n-bit strings x and y (prove the latter property explicitly, it's not enough to write down only the states).

Solutions

1. **(1 point)**

   (a) After applying first CNOT, and then H on the first qubit (in that order!) we have $\frac{1+\sqrt{2}}{\sqrt{6}}|00\rangle + \frac{1-\sqrt{2}}{\sqrt{6}}|10\rangle$

   (b) 0, because $|11\rangle$ has 0 amplitude in the resulting state.

2. **(2 points)**

   (a) Let $x = x_0x_1x_2$ be the 3-bit input.
   Compute $x_0 \oplus x_1$ using one quantum query (for instance by Deutsch-Jozsa algorithm for $n = 1$)
   If $x_0 \oplus x_1 = 0$ then query and output x_0; else query and output x_2.

   This works because if $x_0 \oplus x_1 = 0$ then $x_0 = x_1$ and hence these bits are the majority; and if $x_0 \oplus x_1 = 1$ then $x_0 + x_1 = 1$ and hence x_2 determines the majority.

   (b) Suppose we have a T-query quantum algorithm $\mathcal{A}$ that computes 3-bit majority with success probability 1 for every possible input $x \in \{0,1\}^3$. Then we can use $\mathcal{A}$ to compute the AND of x_0 and x_1 with T queries: run $\mathcal{A}$ on $x = x_0x_10$. Since we know that computing the 2-bit AND function with success probability 1 requires at least 2 queries (Exercise 8.4), we get $T \geq 2$.

   Alternative answer, using polynomial method:
   Suppose we have a T-query algorithm $\mathcal{A}$ that computes 3-bit majority with success probability 1 for every possible input $x \in \{0,1\}^3$. As on page 51/52 of the notes, this induces a 3-variate polynomial $p(x)$ of degree $\leq 2T$ that equals $f(x)$ on every x. You can symmetrize this to a *univariate* polynomial r of degree $\leq 2T$ such that $r(0) = 0$, $r(1) = 0$, $r(2) = 1$, and $r(3) = 1$. Note that the derivative of r has roots in the intervals $[0,1]$ and $[2,3]$, and hence has degree at least 2. Therefore the degree of r itself is at least 3, implying $2T \geq 3$. Since T is an integer, we get $T \geq 2$.
   NB: you cannot argue "p has k roots so degree at least k" here, because that type of argument assumes p is univariate, not multivariate. Also, the adversary method doesn't give the correct lower bound here because of the small constant hidden in the $\Omega(\cdot)$-bound of Eq. (8.1) of the notes.

   (c) Just 1, even for classical algorithms:
   choose one of the 3 indices uniformly at random and query and output that bit. This equals the majority value with probability at least 2/3.
   Clearly there is no 0-query algorithm for a non-constant function, so this 1-query algorithm has the minimal query complexity.

3. **(2.5 points)**

   (a) Run the exact version of Grover s times, once for each possible value of t. For each of those runs, check whether the output-i is a solution. If $|x| \in \{1, \ldots, s\}$, then one of those runs will find a solution with probability 1. The total number of queries is

$$\sum_{k=1}^{s} O(\sqrt{N/k}) + 1 = O(\sqrt{sN}).$$

(b) There exists a version of Grover's algorithm that uses $O(\sqrt{N/s})$ queries, and that (if $|x| > s$) finds a solution with probability at least 1/2 (see p.38 of the lecture notes). Run this algorithm s times, each time checking whether the output is a solution. The probability that none of these s runs finds a solution is $\leq (1/2)^s$. The total query complexity is $s(O(\sqrt{N/s}) + 1) = O(\sqrt{sN})$.

(c) Set $s = \lceil \log(1/\varepsilon) \rceil$. First run the algorithm of (a), then the algorithm of (b). If $|x| \in \{1, \ldots, s\}$ then the algorithm of (a) will find a solution with probability 1. If $|x| > s$ then the algorithm of (b) will find a solution with probability $\geq 1 - 2^{-s} \geq 1 - \varepsilon$. If no solution exists, neither algorithm will return a solution. Both algorithms use $O(\sqrt{sN}) = O(\sqrt{N \log(1/\varepsilon)})$ queries.

4. **(1.5 points)** The input to Simon's problem is $x = (x_0, \ldots, x_{N-1})$, where $N = 2^n$ and each x_i is an n-bit string. Set $G = \mathbb{Z}_2^n = \{0,1\}^n$ (the corresponding QFT is the n-fold Hadamard gate), with subgroup $H = \{0, s\}$, and $f : \{0,1\}^n \to \{0,1\}^n$ is defined as $f(i) = x_i$. Since $f(i) = f(j)$ iff ($i = j$ or $i = j \oplus s$), this f is constant within each coset of H, and distinct on distinct cosets of H. Since $G = \mathbb{Z}_2^n$, the characters are $\chi_j : G \to \{+1, -1\}$ that factor as n characters of $\mathbb{Z}_2$:

$$\chi_j(g) = \prod_{k=1}^{n} \chi_{j_k}(g_k) = \prod_{k=1}^{n} (-1)^{g_k j_k} = (-1)^{g \cdot j}.$$

We have $\chi_j(0^n) = 1$ for all j, and $\chi_j(s) = 1$ iff $s \cdot j = 0 \bmod 2$. Hence

$$H^\perp = \{\chi_j \mid \chi_j(h) = 1 \text{ for all } h \in H\} = \{\chi_j \mid s \cdot j = 0 \bmod 2\}.$$

If you sample uniformly $O(n)$ times from the labels of $H^\perp$, then with high probability you will see $n - 1$ linearly independent j's, all satisfying $s \cdot j = 0 \pmod 2$. From these, using classical Gaussian elimination (mod 2) you can calculate s, thus solving Simon's problem.

5. **(2 points)**

(a) If $x \neq y$, then the polynomial $P_x - P_y$ has degree $\leq n-1$ and is not identically equal to 0, hence it has at most $n - 1$ roots. Accordingly, the probably that a uniformly random $t \in \mathbb{F}_p$ makes $P_x(t)$ and $P_y(t)$ equal, is at most $(n-1)/p \leq 1/3$.

(b) Alice chooses a uniformly random $t \in \mathbb{F}_p$ and sends Bob t and $P_x(t)$, at the expense of $2\lceil \log p \rceil = O(\log n)$ bits of communication. Bob computes $P_y(t)$ from t and his input y, and outputs "equal" if $P_x(t) = P_y(t)$, and outputs "not equal" otherwise. This protocol has success probability 1 if $x = y$, and success probability $\geq 2/3$ if $x \neq y$ (because of part (a)).

(c) For each $x \in \{0,1\}^n$, define the $2\lceil \log p \rceil$-qubit state as follows:

$$|\phi_x\rangle = \frac{1}{\sqrt{p}} \sum_{t \in \mathbb{F}_p} |t\rangle |P_x(t)\rangle.$$

If $x \neq y$ then we have

$$\langle \phi_x | \phi_y \rangle = \frac{1}{p} \sum_{t \in \mathbb{F}_p} (\langle t | \langle P_x(t) |) \cdot (|t\rangle |P_y(t)\rangle) = \frac{1}{p} \sum_{t \in \mathbb{F}_p} \langle P_x(t) | P_y(t) \rangle = \frac{|\{t \mid P_x(t) = P_y(t)\}|}{p} \in [0, 1/3],$$

where the last step is by part (a).