Ph 219b/CS 219b

Exercises
Due: Wednesday 22 February 2006

6.1 Estimating the trace of a unitary matrix

Recall that using an oracle that applies the conditional unitary $\Lambda(U)$,

$$\Lambda(U): \quad \begin{aligned} |0\rangle \otimes |\psi\rangle &\mapsto |0\rangle \otimes |\psi\rangle\ , \\ |1\rangle \otimes |\psi\rangle &\mapsto |1\rangle \otimes U|\psi\rangle \end{aligned} \tag{1}$$

(where U is a unitary transformation acting on n qubits), we can measure the eigenvalues of U. If the state $|\psi\rangle$ is the eigenstate $|\lambda\rangle$ of U with eigenvalue $\lambda = \exp(2\pi i\phi)$, then by querying the oracle k times, we can determine ϕ to accuracy $O(1/\sqrt{k})$.

But suppose that we replace the pure state $|\psi\rangle$ in eq. (1) by the maximally mixed state of n qubits, $\rho = I/2^n$.

a) Show that, with k queries, we can estimate both the real part and the imaginary part of $\mathrm{tr}\,(U)\,/2^n$, the normalized trace of U, to accuracy $O(1/\sqrt{k})$.

b) Given a polynomial-size quantum circuit, the problem of estimating to fixed accuracy the normalized trace of the unitary transformation realized by the circuit is believed to be a hard problem classically. Explain how this problem can be solved efficiently with a quantum computer.

The initial state needed for each query consists of one qubit in the pure state $|0\rangle$ and n qubits in the maximally mixed state. Surprisingly, then, the initial state of the computer that we require to run this (apparently) powerful quantum algorithm contains only a constant number of "clean" qubits, and $O(n)$ very noisy qubits.

6.2 A generalization of Simon's problem

Simon's problem is a hidden subgroup problem with $G = Z_2^n$ and $H = Z_2 = \{0, a\}$. Consider instead the case where $H = Z_2^k$, with generator set $\{a_i, i = 1, 2, 3, \ldots, k\}$. That is, suppose an oracle evaluates a function

$$f : \{0,1\}^n \to \{0,1\}^{n-k} \ , \tag{2}$$

where we are promised that f is 2^k-to-1 such that

$$f(x) = f(x \oplus a_i) \tag{3}$$

for $i = 1, 2, 3, \ldots, k$ (here $\oplus$ denotes bitwise addition modulo 2). Since the number of cosets of H in G is smaller, we can expect that the hidden subgroup is easier to find for this problem than in Simon's ($k = 1$) case.

Find an algorithm using $n - k$ quantum queries that identifies the k generators of H, and show that the success probability of the algorithm is greater than 1/4.

6.3 Query complexity of non-abelian hidden subgroup problems

We have seen that there is an efficient quantum algorithm that solves the hidden subgroup problem for any finitely generated abelian group. What about non-abelian groups? The purpose of this exercise is to show that for any finite group G, the hidden subgroup problem can be solved with polylog($|G|$) queries to the oracle, an exponential improvement over the best classical algorithm.

The oracle evaluates a function

$$f : G \to X \ , \tag{4}$$

from the group G to a set X, that is constant and distinct on the cosets of a subgroup $H \leq G$. The problem is to identify H. The input register contains at least $\log |G|$ qubits ($|G|$ denotes the order of G, the number of elements that it contains), and there are basis states $\{|g\rangle, g \in G\}$ that can be identified with group elements, such that $\langle g'|g\rangle = 0$ for $g \neq g'$. We can query the oracle with a uniform superposition of all group elements, and so prepare the state

$$\frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle \otimes |f(g)\rangle \ . \tag{5}$$

By measuring the output register, we prepare the input register in a randomly selected "coset state" $|gH\rangle$, a uniform superposition of the elements of the coset:

$$|gH\rangle \equiv \frac{1}{\sqrt{|H|}} \sum_{h \in H} |gh\rangle \ . \tag{6}$$

Distinct cosets of H are disjoint, so the coset states corresponding to two distinct cosets are orthogonal.

a) Note that if H and H' are two subgroups of G, then their intersection $H \cap H'$ is also a subgroup of G. Show that if the intersection of cosets $g'H' \cap gH$ is not empty, then it contains $|H \cap H'|$ elements of G.

b) Recall that if $H' \le H$, then $|H'|$ divides $|H|$. Let P_H denote the orthogonal projection onto the linear span of the coset states $\{|gH\rangle, g \in G\}$. Show that

$$\|P_{H'}|gH\rangle\|^2 = \frac{|H \cap H'|}{|H'|} \ . \tag{7}$$

Conclude that $P_{H'}|gH\rangle = |gH\rangle$ for $H' \le H$, and that for $H' \not\le H$,

$$\|P_{H'}|gH\rangle\| \le \frac{1}{\sqrt{2}} \ . \tag{8}$$

With k queries we can prepare a tensor product of randomly selected coset states

$$|\psi\rangle = |g_1H\rangle \otimes |g_2H\rangle \otimes \cdots \otimes |g_kH\rangle \ . \tag{9}$$

Let $P_H^{(k)} \equiv P_H^{\otimes k}$ denote the orthogonal projector onto the linear span $V_H^{(k)}$ of all such product states. Note that for $H' \not\le H$,

$$\|P_{H'}^{(k)}|\psi\rangle\| \le \frac{1}{2^{k/2}} \ . \tag{10}$$

Thus if the actual hidden subgroup is H, and we make an orthogonal measurement $M_{H'}^{(k)}$ that projects onto either $V_{H'}^{(k)}$ (the positive outcome) or its orthogonal complement $V_{H'}^{(k)\perp}$ (the negative outcome), then we obtain the positive outcome with probability one for $H' \le H$, and we obtain the positive outcome with probability no greater than 2^{-k} for $H' \not\le H$.

We can make a list $H_1, H_2 \ldots, H_R$ of the candidate hidden subgroups starting with the largest subgroups, such that no H_r on the list is a subgroup of another H_s for $s > r$; then we perform a series of tests to identify the hidden subgroup by first performing the measurement $M_{H_1}^{(k)}$, then $M_{H_2}^{(k)}$, continuing until a positive outcome is obtained for the first time. The first H' for which the measurement yields a positive outcome is identified as the hidden subgroup.

To check that this algorithm really works, we need to verify that the measurements that yield negative outcomes do not disturb the state $|\psi\rangle$ very much. If the actually hidden subgroup is H_r (the rth one listed), then the probability that the algorithm successfully identifies H_r is

$$P_{\text{success}} = \|P_{H_r}\left(I - P_{H_{r-1}}\right)\left(I - P_{H_{r-2}}\right)\cdots\left(I - P_{H_1}\right)|\psi\rangle\|^2 \qquad (11)$$

c) Note that

$$\|A|\varphi\rangle - A(I-B)|\varphi\rangle\| = \|AB|\varphi\rangle\| \leq \|B|\varphi\rangle\| \qquad (12)$$

for $\|A\|_{\text{sup}} \leq 1$; applying eq. (12) repeatedly $r-1$ times, conclude that

$$P_{\text{success}} \geq \left(1 - \frac{r-1}{2^{k/2}}\right)^2 . \qquad (13)$$

We see that the algorithm has constant success probability for $k = O(\log R)$, where R is the number of candidates for the hidden subgroup. In fact, since any subgroup of G can be generated by a set of at most $n = \log_2 |G|$ elements of G (you are not asked to prove this), and there are fewer than $|G|^n$ ways to choose n elements of $|G|$, the number of subgroups of G (and hence R) is less than 2^{n^2}. Therefore $O(\log^2 |G|)$ queries suffice to solve the hidden subgroup problem.

However ... in contrast to the solution to the abelian hidden subgroup problem, our algorithm runs in exponential time, because we may have to perform R measurements after the queries are completed. So far, polynomial time (in $\log |G|$) algorithms that solve the hidden subgroup problem for non-abelian G are known in only a few special cases.

6.4 BQP is contained in PP

We have seen that **BQP** (the class of decision problems that can be solved efficiently by a quantum computer) is contained in the classical complexity class **PSPACE** (the decision problems that can be solved using a polynomial-size memory). The purpose of this problem is to establish an inclusion that is presumed to be stronger: **BQP** is contained in **PP**. A decision problem is in **PP** ("probabilistic polynomial time") if it can be solved in polynomial time by a randomized classical computation with probability of success greater than 1/2. (In contrast to the class **BPP**, the success probability is not required to exceed 1/2 by a positive constant independent of the input size.)

When a decision problem is solved by a quantum computer, one particular qubit may be designated as the "answer qubit" — the qubit that is measured to decide the answer. If the quantum circuit builds the unitary transformation U, which acts on the input state $|0\rangle$, then the probability distribution governing the answer bit x can be expressed as

$$P(x) = \sum_y |\langle x, y|U|0\rangle|^2 \ , \tag{14}$$

where $|y\rangle$ denotes a basis state for the "junk" output qubits that are not measured. If the problem is in **BQP**, then there is a polynomial-size uniform quantum circuit family (where the circuit depends on the input to the problem) such that $P(x) \geq 2/3$ when x is the correct value of the output.

a) Show that if a problem is in **BQP** then there is a polynomial-size uniform circuit family (where the circuit depends on the input to the problem) such that $|\langle 0|U|0\rangle|^2 \geq 2/3$ if the correct output is 0 and $|\langle 0|U|0\rangle|^2 < 1/3$ if the correct output is 1. **Hint**: We want to avoid summing over the state of the unmeasured "junk." Recall the trick we used to remove the junk produced by a reversible classical circuit.

If we want to simulate on a classical computer the quantum computation that solves the problem, then, it suffices to estimate the single matrix element $|\langle 0|U|0\rangle|$ to reasonable accuracy.

Now recall that we saw in Exercise 5.3 that the quantum gate set $\{\boldsymbol{H}, \Lambda^2(\boldsymbol{X})\}$, where $\boldsymbol{H}$ denotes the Hadamard gate and $\Lambda^2(\boldsymbol{X})$ is the Toffoli gate, is universal for quantum computation. The Toffoli gate is classical, but the Hadamard gate takes the computational basis states of a qubit to superpositions of basis states:

$$\boldsymbol{H} : |x\rangle \mapsto \frac{1}{\sqrt{2}} \sum_{y\in\{0,1\}} (-1)^{xy}|y\rangle \ . \tag{15}$$

Note that, since $\boldsymbol{H}^2 = I$, we are free to insert a pair of Hadamard gates acting on all qubits following each Toffoli gate, without changing anything.

Suppose that U is expressed as a circuit constructed from Toffoli gates and Hadamard gates, where the circuit contains h Hadamard gates, which we label by $i = 0, 1, 2, \ldots, h-1$. By inserting the partition of unity $I = \sum_{x_i \in \{0,1\}} |x_i\rangle\langle x_i|$ following each Hadamard gate, we may write the matrix element $\langle 0|U|0\rangle$ as a sum of 2^h terms, with each term arising from a "computational path" indexed by the bit string $x = x_{h-1}x_{h-2}\ldots x_1x_0$. Each term has absolute value $2^{-h/2}$, arising from the factor $2^{-1/2}$ that accompanies each Hadamard gate, and a phase ± 1 depending on x that counts modulo 2 the number of Hadamard gates for which the input and output both have the value 1. Note that each output bit from every Toffoli gate, and therefore the input to each Hadamard gate, can be expressed as a polynomial in the $\{x_i\}$ that depends on the circuit.

b) Show that, if U is a unitary transformation constructed from a circuit of size L, built from Toffoli and Hadamard gates, then there is a polynomial function $\phi(x)$ of $h \le 2L$ binary variables $\{x_{h-1}, x_{h-2}, \ldots x_1, x_0\}$ such that

$$\langle 0|U|0\rangle = \frac{1}{\sqrt{2^h}}\left(N_0 - N_1\right) \ , \tag{16}$$

where N_0 is the number of values of x for which $\phi(x) = 0$ (mod 2) and N_1 is the number of values of x for which $\phi(x) = 1$ (mod 2). Furthermore, $\phi(x)$ is of degree at most three, is a sum of at most $2h$ monomials, and can be computed efficiently from a description of the circuit.

Thus, if the circuit is polynomial size, the function $\phi(x)$ can be evaluated efficiently for any of the 2^h possible values of its input x. The difference $N_0 - N_1$ is $O(\sqrt{2^h})$, and we can simulate the quantum circuit (distinguishing $|\langle 0|U|0\rangle|^2 \ge 2/3$ from $|\langle 0|U|0\rangle|^2 < 1/3$) if we can estimate $N_0 - N_1$ to $O(\sqrt{2^h})$.

Up until now, the computational problems we have encountered have typically involved determining whether a solution to an equation exists, exhibiting a solution, or verifying the correctness of a solution.

Here we have encountered a problem that appears to be intrinsically harder: *counting* (approximately) the number of solutions.

We could attempt to do the counting by a randomized computation, evaluating $\phi(x)$ for a sample of randomly selected inputs. Unfortunately, the number of inputs mapped to 0 and to 1 are nearly in balance, which makes it hard to estimate $N_0 - N_1$. We can think of $\phi(x)$ as a coin with an exponentially small bias; determining the basis to reasonable accuracy requires an exponentially large number of trials.

But the criterion for inclusion in the class **PP** is weaker than that — it is enough to be able to distinguish $|\langle 0|U|0\rangle|^2 \geq 2/3$ from $|\langle 0|U|0\rangle|^2 < 1/3$ with probability $1/2 + \delta$ as long as δ is positive, even if it is exponentially small. This can be achieved with a polynomial number of trials. Therefore, **BQP** $\subseteq$ **PP**.

The class **PP** is certainly contained in **PSPACE**, because we can determine the probability of acceptance for a randomized computation by simulating all of the possible computational paths that occur for all possible outcomes of the coin flips. There may be an exponentially large number of paths, but we can run through the paths one at a time, while maintaining a count of how many of the computations have accepted. This can be done with polynomial space.