Stanford University — CS259Q: Quantum Computing Handout 7
Luca Trevisan October 16, 2012

Lecture 7

Updated 11/6/2012

In which we describe the quantum Hadamard Fourier transform and its application to a simple Boolean "period-finding" problem, which is a simplified version of Shor's factoring algorithm, containing all the quantum ideas and none of the number theory.

1 The Hadamard Transform

In this section we describe a variant of the discrete Fourier transform that is applicable to functions with Boolean inputs. It is usually called the S transform, where S is an arbitrary non-empty subset of $\{\text{Hadamard}, \text{Rademacher}, \text{Walsh}\}$; we will call it the Hadamard transform.

We consider the 2^n-dimensional vector space of functions

$$f : \{0,1\}^n \to \mathbb{C}$$

with the inner product

$$\langle f, g \rangle := \sum_x f(x)\overline{g(x)}$$

For every $s \in \{0,1\}^n$ we define the function

$$\chi_s(x) := \frac{1}{2^n}(-1)^{\sum_i x_i s_i}$$

It is easy to see that these 2^n functions form an orthonormal basis for the vector space of functions $f : \{0,1\}^n \to \mathbb{C}$. For every s, t, we have

$$\langle \chi_s, \chi_t \rangle = \frac{1}{2^n} \sum_x (-1)^{\sum_i (s_i + t_i) x_i}$$

If $s = t$, then $\sum_i (s_i + t_i) x_i$ is always even and so

$$\langle \chi_s, \chi_t \rangle = \frac{1}{2^n} \sum_x 1 = 1$$

If $s \neq t$, then let j be one coordinate such that $s_j \neq t_j$, and note that

$$\langle \chi_s, \chi_t \rangle = \mathop{\mathbb{P}}_{x \sim \{0,1\}^n} \left[\sum_i (s_i + t_i) x_i \equiv 0 \pmod 2 \right] - \mathop{\mathbb{P}}_{x \sim \{0,1\}^n} \left[\sum_i (s_i + t_i) x_i \equiv 1 \pmod 2 \right] = 0$$

where the fact that the two probabilities are equal, and hence their difference is zero, follows from the fact that we can get a bijection between the strings x for which $\sum_i (s_i + t_i) x_i$ is even and the strings x for which the expression is odd by matching strings that differ only in the j-th coordinate.

Having proved that the functions χ_s form an orthonormal basis, we have that every function $f : \{0,1\}^n \to \mathbb{C}$ can be written as a linear combination

$$f(x) = \sum_s \hat{f}(s) \chi_s(x)$$

where

$$\hat{f}(s) = \langle f, \chi_s \rangle = \frac{1}{\sqrt{2^n}} f(x) (-1)^{\sum_i x_i s_i}$$

2 Quantum Computation of the Hadamard Transform

By Parseval's identity, the transformation

$$\sum_x f(x) |x\rangle \to \sum_s \hat{f}(s) |s\rangle$$

is unitary.

It is not hard to see that it can also be realized by a linear-size quantum circuit that simply applies a Hadamard gate to each input qubit. To verify this claim it is sufficient to observe that the Hadamard transform realizes the operation:

$$|x\rangle \to \frac{1}{\sqrt{2}^n} \sum_s (-1)^{\sum_i x_i s_i} |s\rangle$$

$$= \bigotimes_{i=1}^{n} \frac{1}{\sqrt{2}} \left(|0\rangle + (-1)^{x_i} |1\rangle \right) = Hx_1 \otimes Hx_2 \otimes \cdots \otimes Hx_n$$

3 Simon's Algorithm

In this section, we describe and analyze a quantum algorithm that solves the Boolean version of the problem of finding the period of a periodic function. Although the problem is rather artificial, the algorithm makes all the steps and contains all the quantum ideas of Shor's algorithm for factoring, without all the number theory, the trigonometry, and the need to deal with roundings and approximations. For this reason, understanding Simon's algorithm is an excellent preparation to Shor's algorithm.

The problem that we want to solve is as follows: we are given a boolean function $f : \{0,1\}^n \to \{0,1\}^n$ which is efficiently computable (say, by a circuit of size $S(n) = n^{O(1)}$) and that is 2-to-1, meaning that every possible output has precisely two preimages; furthermore, we are promised that there is a non-zero boolean string r such that for every x we have $f(x) = f(x \oplus r)$. Our goal is to find collisions of f or, equivalently, the string r.

Before proceeding, we recall a fact from our treatment of classical circuits and quantum circuits: suppose that $g : \{0,1\}^n \to \{0,1\}^m$ is a function computable by a classical circuit of size S. Then there is a quantum circuit that operates on $n + m + O(S)$ qubits, of size $O(S)$, made of U_{CNOT} gates and one-qubit gates, that computes a unitary transformation U_g such that for all $x \in \{0,1\}^n$ we have

$$U_g(|x\rangle|0\cdots0\rangle|0\cdots0\rangle) = |x\rangle|g(x)\rangle|0\cdots0\rangle$$

where the first part of the input is of length n, the second of length m, and the third of length $O(S)$.

We now turn to Simon's algorithm, which proceeds as follows:

1. **Create the quantum state $\frac{1}{\sqrt{2^n}} \sum_x |x\rangle|f(x)\rangle$**

   Let U_f be a unitary transformation on $\ell = n + n + O(S(n))$ bits that maps $|x\rangle|0\cdots0\rangle|0\cdots0\rangle$ to $|x\rangle|f(x)\rangle|0\cdots0\rangle$. We construct a circuit over ℓ qubits that first applies Hadamard gates to each of the first n inputs. After these operations, starting from the input $|0^\ell\rangle$ we get $\frac{1}{\sqrt{2^n}}|x\rangle|0^{\ell-n}\rangle$. Then we apply U_f, which gives us the state $\frac{1}{\sqrt{2^n}}|x\rangle|f(x)\rangle|0^{\ell-2n}\rangle$. From this point on, we ignore the last $\ell - 2n$ wires.

2. **Measure the last n bits of the state**

   The outcome of this measurement will be a possible output $f(z)$ of $f(\cdot)$, and, for each such possible outcome, the residual state will be

$$\frac{1}{\sqrt{2}}|z\rangle|f(z)\rangle + \frac{1}{\sqrt{2}}|z+r\rangle|f(z)\rangle$$

From this point on we ignore the last n bits of the state because they have been fixed by the measurement.

3. **Apply the Hadamard transform to the first n bits**

   The state becomes

$$\frac{1}{\sqrt{2^{n+1}}}\sum_s \left((-1)^{\sum_i s_i z_i} + (-1)^{\sum_i s_i(z_i+r_i)}\right)|s\rangle$$

   Now we see that the amplitude of $|s\rangle$ in the above state is $\frac{1}{\sqrt{2^{n-1}}}$ if

$$\sum_i s_i z_i \equiv \sum_i s_i(z_i+r_i) \pmod 2$$

   which is equivalent to

$$\sum_i s_i r_i \equiv 0 \pmod 2$$

   and it is zero otherwise.

4. **Measure the first n bits.**

   The measurement will give us a random element of the set of strings s such that

$$\sum_i s_i r_i \equiv 0 \pmod 2$$

   If we think of s and r as vectors in the n-dimensional vector space $\mathbb{F}_2^n$, then we get a linear equation about r.

Now we repeat the above algorithm until we find a collection of strings $s^{(j)}$ such that for each j we have

$$\sum_i s_i^{(j)} r_i \equiv 0 \pmod 2$$

and such that the $s^{(j)}$ span the whole space of vectors s such that $\sum_i s_i r_i \bmod 2 = 0$. This is an $(n-1)$-dimensional space, so we need to find $n-1$ $s^{(j)}$ that are linearly independent. It is easy to see that the algorithm only needs to be executed $O(n)$ times on average until this condition is satisfied.

Once we have $n-1$ linearly independent linear equations in the unknown r, and we can find a 1-dimensional space of solutions via Gaussian elimination. Such a space will contain just two solutions, $(0 \cdots 0)$ and r, so by picking the nonzero solution we find r.