QUANTUM COMPUTATION - FALL 2006 - EXERCISE 4

Due: Wednesday 11/1/06. Submition in triplets, to the TIRGUL.

1. Do We Really Need Complex Numbers?

Prove that we can assume w.l.o.g that all amplitudes in a quantum computation are real numbers. **Guidence:** Show that any quantum circuit on n qubits that uses t two-qubit gates can be simulated exactly by a quantum circuit on $n+1$ qubits, that uses at most t three-qubit gates. (Hint: use the extra qubit to decide whether we are in the Imaginary or Real world.)

2. Fourier Transform

2.1. let a, b such that $a|q$ and $b|q$ and $\gcd(a,b) = 1$. Let $S = \{0 \leq x < q \text{ s.t. } a|x \text{ or } b|x\}$, and define the state

$$|\phi\rangle = \frac{1}{\sqrt{|S|}} \sum_{x \in S} |x\rangle .$$

What is the Fourier transform modulo q of $|\phi\rangle$?

2.2. Consider the FT over Z_{3^n}. Design a quantum circuit that works with qutrits (qubits with three states, $|0\rangle, |1\rangle, |2\rangle$) and recursively computes the FT over this group.

3. Shor's Algorithm

A student suggested the following idea. In the presentation in class of Shor's algorithm, in the simple case (where r devided Q) we pick many random k's, $k_1, k_2, \ldots$, where we have $k_i = m_i Q/r$. We claimed that as long as one of the m_i's is coprime with r, we are OK. The student claimed that he knows what to do even if none of the m_i's are coprime with r, but just two of them are coprime to each other.

3.1. Complete the student's idea: give a way to find r if you are given two k's, such that the relevant m's are coprime with each other.

3.2. Show how to use this to make Shor's algorithm in the simple case more efficient, using the following fact: The probability for two independently chosen random numbers smaller than an integer n to be coprime converges to $\frac{6}{\pi^2}$ as n goes to infinity.

3.3. Can you use this idea in some way to improve the complexity of the general case of Shor's algorithm? Explain your thoughts. (This is more of an open discussion question.)

4. Grover's Algorithm

4.1. You are given access (through an oracle) to a database of N items with M marked items. Provide an algorithm that finds a marked item in $O(\sqrt{N})$ queries, even if you don't know M. (Here you are requested to complete the details of the sketch given in class. Recall that the main idea was that we can use a random number of iterations).

4.2. Build on the idea of the previous subquestion, and design an algorithm that would find a marked item in $O(\sqrt{N/M})$ queries. Guidance: The value of M lies between 2^j and 2^{j+1} for some j. Show that if you knew j you could perform the algorithm, and then use binary search to look for j.