Homework Assignment #4

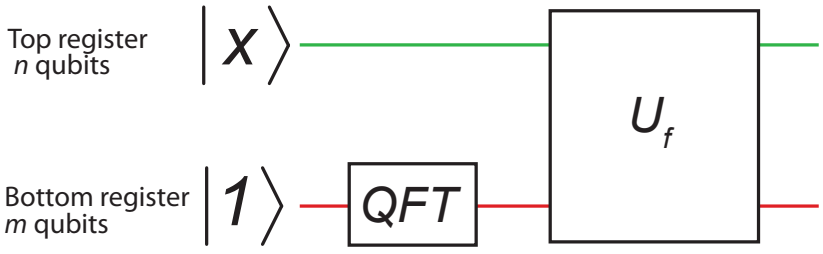
Due at start of class on Dec. 13, 2013

Problem 1: *Quantum Fourier Transform.*

- Write the 8×8 matrix (in the computational basis) corresponding to the quantum Fourier transform on a 3-qubit register.
- Show that this matrix is unitary.
- Draw the quantum circuit that implements this QFT.

Problem 2: *Generalized quantum kick-back*

In the lectures, we have seen how the Deutsch and Bernstein-Vazirani quantum games exploit quantum kick-back to efficiently extract properties of n-to-1 bit boolean functions. In this problem, we generalize quantum kick-back to n-to-m bit boolean functions encoded in unitary functions as usual: $U_f|x\rangle|y\rangle = |x\rangle|(y + f(x)) \bmod M\rangle$ for computational states $|x\rangle$ and $|y\rangle$ in the top and bottom registers, respectively. Consider the circuit below.



- What is the state of the bottom register after application of the m-bit QFT on the initial state $|1\rangle = |00000\ldots1\rangle$.
- Now apply U_f, with the top register starting in a computational state $|x\rangle$. What is the combined state of the top and bottom registers immediately after U_f? Show that this this state can be rewritten as

$$e^{-i2\pi f(x)/M}|x\rangle \otimes U_{QFT}|1\rangle,$$

  where $M = 2^m$. Evidently, the top and bottom registers are not entangled, and $f(x)$ is encoded in the quantum phase of the probability amplitude!
- Finally, consider the case that the top register is initialized in the maximal superposition state $\frac{1}{\sqrt{N}}(|0\rangle + \cdots + |N-1\rangle)$. As usual, $N = 2^n$. What will be the final state after application of U_f?

Problem 3: *Breaking RSA.*
In this exercise, we will break RSA by period finding. N will be small enough that we will find periods by brute force.

- List the integers $a < N$ that are co-prime with N. Let us pick one of these integers: let us agree to all "randomly" pick $a = 8$.
- Compute $8^0 \text{mod} 21, 8^1 \text{mod} 21, 8^2 \text{mod} 21 \ \ldots$ until you discover the period r of $f(x) = 8^x \text{mod} 21$.
- Find the greatest common denominator (gcd) between, $8^{r/2} + 1$ and 21. Check whether the result is a prime factor of 21.
- Similarly, find the gcd between $8^{r/2} - 1$ and 21.
- Repeat the same steps as above, but this time using $a = 10$;

Problem 4: *Breaking RSA with Shor's algorithm.*
Now we will go through the steps of Shor's algorithm in order to find the period r of $8^x \text{mod} 21$. For simplicity, lest us use just four qubits for the top register (this is enough for our choice of $a = 8$.)
Note: Please use decimal instead of binary notation for the states of each register. For instance, write $|0011\rangle$ *simply as* $|3\rangle$.

- Initialize each register to $|0\rangle$.
- Apply the Hadamard gate to each qubit in the first register.
- Add $8^x \text{mod} 21$ to the second register, where x is the state of the first register.
- Rewrite this state so you group all terms with identical $f(x)$ — observe the periodicity in the amplitudes which emerges, and observe also that you cannot efficiently reveal the period by any measurement.
- Apply the Quantum Fourier Transform to the top register.
- Measure the final state of the top register. What are the possible measurement outcomes, and how do they relate to r?
- What fraction of the times that you run the algorithm above will you successfully find the factors of 21?
- What complication occurs for $a = 10$? Discuss how Shor deals with it. *Please answer this question in just a few sentences.*

Problem 5: (EXTRA CREDIT) *Tensor product matrices and product states.*
A general tensor product of two one qubit unitaries A and B is given by

$$A \otimes B = \begin{pmatrix} A_{00}\begin{pmatrix} B_{00} & B_{01} \\ B_{10} & B_{11} \end{pmatrix} & A_{01}\begin{pmatrix} B_{00} & B_{01} \\ B_{10} & B_{11} \end{pmatrix} \\ A_{10}\begin{pmatrix} B_{00} & B_{01} \\ B_{10} & B_{11} \end{pmatrix} & A_{11}\begin{pmatrix} B_{00} & B_{01} \\ B_{10} & B_{11} \end{pmatrix} \end{pmatrix}.$$

1. Show that a matrix of this form always takes product states to product states.
2. Proof that the CNOT cannot be written in this form by comparing the matrix entries with the general tensor product matrix above. (Note that the CNOT gate can take some product states to entangled states).
3. Show that the SWAP gate always takes a product state to another product state.
4. Can you write the SWAP gate in the form $A \otimes B$? If so give your solution for A and B. If not give a proof.