CS 410/510 Introduction to Quantum Computing

Homework 4

Portland State U, Spring 2020 *04/26/2020*

Lecturer: Fang Song *Due: 11:59pm PDT,* ***05/10/2020***

Instructions. This problem set contains 20 pages (including this cover page) and 5 questions. Problems marked with "[**G**]" are required for 510 students. Students enrolled in 410 will get bonus points for solving them. A random subset of problems will be graded.

Your solutions will be graded on *correctness* and *clarity*. You should only submit work that you believe to be correct, and you will get significantly more partial credit if you clearly identify the gap(s) in your solution. It is good practice to start any long solution with an informal (but accurate) summary that describes the main idea.

You need to submit a PDF file via Gradescope before the deadline. Either a clear scan of you handwriting or a typeset document is accepted. You will get 5 bonus points for typing in LaTeX (Download and use the accompany TeX file).

You may collaborate with others on this problem set. However, you must ***write up your own solutions*** and ***list your collaborators*** for each problem.

1. (Measurement in Simon's algorithm) Our analysis in class assumes that we measure the bottom n qubits right after the query to the quantum black-box. In this problem, we investigate different timing for this measurement.

   (a) (5 points) Suppose we do not measure immediately, and move on to apply Hadamard to each of the top n qubits. Write down the quantum state of the entire system ($2n$ qubits) after this step.

(b) (5 points) Then we measure the bottom n qubits in the computational basis. Suppose the outcome is $a \in \{0,1\}^n$, describe the posterior state. Is it the same as in the analysis where the measurement happens before the Hadamard's?

(c) (5 points) Consider another continuation of part (a). We do not measure the bottom n qubits at all, and proceed to measure the top n qubits only. Describe the result of the measurement. (Hint. For a state $\sum_z |z\rangle_A |\phi_z\rangle_B$, measuring system A will result in z with probability $\||\psi_z\rangle\|^2$.)

2. (Spectral theorem) Let $U = (v_1, \ldots, v_n)$ be a unitary matrix and each $v_i \in \mathbb{C}^n$.
   (a) (5 points) Show that $\{v_1, \ldots, v_n\}$ form an orthonormal basis of $\mathbb{C}^n$.

(b) (5 points) Show that the eigenvalues of any unitary U are of the form $e^{2\pi i\theta}$ for some $\theta \in [0, 1)$.

(c) (5 points) Show that the eigenvectors corresponding to distinct eigenvalues are orthogonal. (N.B. You can also verify that for each distinct eigenvalue λ, the set $S_\lambda := \{v : Uv = \lambda v\}$ is a subspace, which is called the λ-eigenspace.)

3. (Simulate classical circuit) The function $\mathrm{EQ} : \{0,1\}^3 \to \{0,1\}$ determines whether its three input bits are equal, namely

$$\mathrm{EQ}(a,b,c) \mapsto \begin{cases} 1 & \text{if } a = b = c \\ 0 & \text{otherwise} \end{cases}.$$

(a) (6 points) Show how to implement the OR gate and the duplicate gate (DUP) in a reversible way using Toffoli gate. (Note: we showed this for AND gate in class, which you can use as a building block. You may also use ancilla bits initialized to either 0 or 1.)

(b) (5 points) Show how to compute the function EQ using AND, OR, NOT, and DUP gates.

(c) (5 points) Show how to compute the function EQ reversibly using Toffoli gates. You may use gates other than Toffoli gates provided you explain how to implement any such gates using Toffoli gates.

(d) (4 points) Turn your reversible circuit into a quantum circuit that implements the unitary $U_{\mathrm{EQ}} : |x\rangle|y\rangle \mapsto |x\rangle|\mathrm{EQ}(x) \oplus y\rangle$.

4. (Square root of a unitary)

   (a) (5 points) Let U be a unitary operation with eigenvalues ± 1. Let P_0 be the projection onto the $+1$ eigenspace of U and let P_1 be the projection onto the -1 eigenspace of U. Let $V = P_0 + iP_1$. Show that $V^2 = U$.

(b) (5 points) Give a circuit of 1- and 2-qubit gates and controlled-U gates with the following behavior (where the first register is a single qubit):

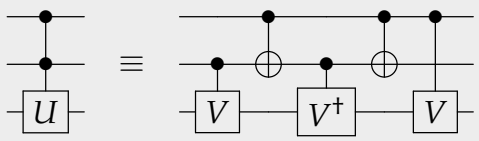
$$|0\rangle|\psi\rangle \mapsto \begin{cases} |0\rangle|\psi\rangle & \text{if } U|\psi\rangle = |\psi\rangle \\ |1\rangle|\psi\rangle & \text{if } U|\psi\rangle = -|\psi\rangle \end{cases}.$$

(c) (5 points) Give a circuit of 1- and 2-qubit gates and controlled-U gates that implements V, and show that it has the desired behavior. Your circuit may use ancilla qubits that begin and end in the $|0\rangle$ state.

(d) (5 points) As an application, the square root of a unitary operation U is useful to implement a controlled-controlled version of U:

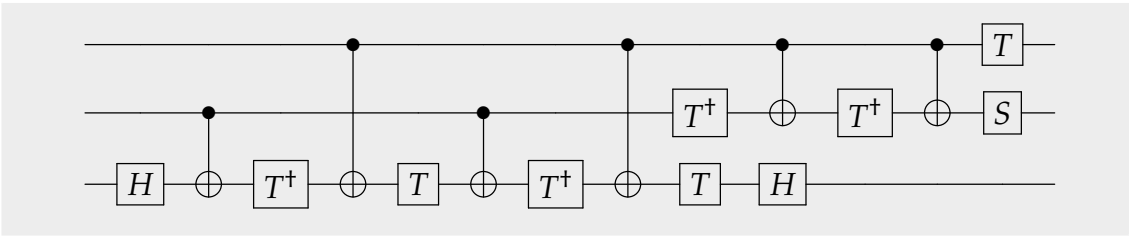
$$CC-U : |a\rangle|b\rangle|\psi\rangle \mapsto |a\rangle|b\rangle U^{a\cdot b}|\psi\rangle\,.$$

Namely U is applied to the target ψ iff. both control bits $a = b = 1$. Let $V^2 = U$. Prove the following circuit identity.



(e) (5 points (bonus)) Show how to implement $V = \sqrt{X}$ in QISKIT. (Note: this gives you a way to implement Toffoli following part (d).)

(f) (5 points) Determine the behavior of the following quantum circuit by implementing it (in IBM QISKIT or other tools of your choice).



5. (Errors in randomized algorithms) Suppose you want to write a computer program C to compute a Boolean function $f : \{0,1\}^n \to \{0,1\}$, mapping n bits to 1 bit. If C is a deterministic algorithm, then "C successfully computes f" has a clear meaning that that $C(x) = f(x)$ for all inputs $x \in \{0,1\}^n$. But what if C is a probabilistic algorithm?

   (a) (8 points) The best thing is if C is a *zero-error* algorithm with failure probability p. Namely

   - on every input x, the output of $C(x)$ is either $f(x)$ or $\perp$ (denoting failure).
   - on every input x we have $\Pr[C(x) = \perp] \leq p$ (NB. the probability is only over the internal randomness of C, not the random choice of x.).

   i) If you have a zero-error algorithm C for f with failure probability 90%, show how to convert it to a zero-error algorithm C' with failure probability at most 2^{-500}. The "slowdown" should only be a factor of a few thousand.

   ii) Alternatively, show how to convert C to an algorithm C'' for f which: (i) always outputs the correct answer, meaning $C''(x) = f(x)$ for all x; (ii) has expected running time only a few powers of 2 worse than that of C. (Hint: look up the mean of a geometric random variable.)

(b) (5 points) The second best thing is if C is a one-sided error algorithm for f, with failure probability p. There are two kinds of such algorithms, "no-false-positives" and "no-false-negatives". For simplicity, let's just consider "no false-negatives" (the other case is symmetric);

- on every input x, the output $C(x)$ is either 0 or 1;
- on every input x such that $f(x) = 1$, the output $C(x)$ is also 1;
- on every input x such that $f(x) = 0$, we have $\Pr[C(x) = 1] \leq p$.

Show how to convert a no-false-negatives algorithm C for f with failure probability 90% to another no-false-negatives algorithm C' for f with failure probability at most 2^{-500}. The "slowdown" should only be a factor of a few thousand.

(c) (5 points (bonus)) The third possibility (which is rare in practice) is if C is a two-sided error algorithm for f, with failure probability p. Namely,

- on every input x, the output $C(x)$ is either 0 or 1.
- on every input x, we have $\Pr[C(x) \neq f(x)] \leq p$.

If you have a two-sided error algorithm C for f with failure probability 40%, show how to convert it to a two-sided error algorithm C' for f with failure probability at most 2^{-500}. The "slowdown" should only be a factor of a few dozen thousand. (Hint: look up the Chernoff bound.)