Quantum Algorithms Exercises

Taken from Ronald de Wolf's *Quantum Computing* lecture notes

1. Construct a CNOT gate from two Hadamard gates and one controlled-Z (the controlled-Z gate maps $|11\rangle \mapsto -|11\rangle$ and acts like the identity on the 3 other basis states).

2. Prove the *quantum no-cloning theorem*: there does not exist a 2-qubit unitary U that maps

$$|\phi\rangle|0\rangle \mapsto |\phi\rangle|\phi\rangle$$

for every qubit $|\phi\rangle$. *Hint: Consider what U has to do when $|\phi\rangle = |0\rangle$, when $|\phi\rangle = |1\rangle$, and when $|\phi\rangle$ is a superposition of these two.*

3. Prove that an EPR-pair $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ is an *entangled* state, i.e., that it cannot be written as the tensor product of two separate qubits.

4. Use Shor's algorithm to find the period of the function $f(a) = 7^a \bmod 10$, using a Fourier transform over $q = 128$ elements. Write down all intermediate superpositions of the algorithm for this case (don't just copy the general expressions form the notes). You may assume you're lucky, so the first run of the algorithm already gives a $b = cq/r$ where c is coprime with r.

5. Let $x = x_0 \dots x_{N-1}$ be a sequence of distinct integers, where $N = 2^n$. We can query these in the usual way, i.e., we can apply unitary $O_x : |i, 0\rangle \mapsto |i, x_i\rangle$, as well as its inverse. The *minimum* of x is defined as $\min\{x_i \mid i \in \{0, \dots, N-1\}\}$. Give a quantum algorithm that finds (with probability $\geq 2/3$) an index achieving the minimum, using at most $O(\sqrt{N} \log N)$ queries to the input. *Hint: Start with $m = x_i$ for a random i, and repeatedly use Grover's algorithm to find an index j such that $x_j < m$ and update $m = x_j$. Continue this until you can find no element smaller than m, and analyze the number of queries of this algorithm. You are allowed to argue about this algorithm on a high level (i.e., things like "use Grover to search for a j such that..." are OK), no need to write out complete circuits.*

6. Consider an undirected graph $G = (V, E)$, with vertex set $V = \{1, \dots, n\}$ and edge-set E. We say G is *connected* if, for every pair of vertices $i, j \in V$, there is a path between i and j in the graph. The *adjacency matrix* of G is the $n \times n$ Boolean matrix M where $M_{ij} = 1$ iff $(i, j) \in E$ (note that M is a symmetric matrix because G is undirected). Suppose we are given input graph G in the form of a unitary that allows us to query whether an edge (i, j) is present in G or not:

$$O_M : |i, j, b\rangle \mapsto |i, j, b \oplus M_{ij}\rangle.$$

   (a) Assume G is connected. Suppose we have a set A of edges which we already know to be in the graph (so $A \subseteq E$; you can think of A as given classically, you don't have to query it). Let $G_A = (V, A)$ be the subgraph induced by only these edges, and suppose G_A is not connected, so it consists of $c > 1$ connected components. Call an edge $(i, j) \in E$

"good" if it connects two of these components. Give a quantum algorithm that finds a good edge with an *expected* number of $O(n/\sqrt{c-1})$ queries to M.

(b) Give a quantum algorithm that uses at most $O(n^{3/2})$ queries to M and decides (with success probability at least $2/3$) whether G is connected or not. This result is due to [2].

(c) Show that classical algorithms for deciding (with success probability at least $2/3$) whether G is connected, need to make $\Omega(n^2)$ queries to M.

7. Let A, B, and C be $n \times n$ matrices with real entries. We'd like to decide whether or not $AB = C$. Of course, you could multiply A and B and compare the result with C, but matrix multiplication is expensive (the current best algorithm takes time roughly $O(n^{2.38})$).

(a) Give a classical randomized algorithm that verifies whether $AB = C$ (with success probability at least $2/3$) using $O(n^2)$ steps, using the fact that matrix-vector multiplication can be done in $O(n^2)$ steps. *Hint: Choose a uniformly random vector $v \in \{0,1\}^n$, calculate ABv and Cv, and check whether these two vectors are the same. This result is by Freivalds [3].*

(b) Show that if we have query-access to the entries of the matrices (i.e., oracles that map $i, j, 0 \mapsto i, j, A_{i,j}$ and similarly for B and C), then any classical algorithm with small error probability needs at least n^2 queries to detect a difference between AB and C. *Hint: Consider the case $A = I$.*

(c) Give a quantum walk algorithm that verifies whether $AB = C$ (with success probability at least $2/3$) using $O(n^{5/3})$ queries to matrix-entries. *Hint: Modify the algorithm for collision-finding: use a random walk on the Johnson graph $J(n, r)$, where each vertex corresponds to a set $R \subseteq [n]$ of size r, and that vertex is marked if there are $i, j \in R$ such that $(AB)_{i,j} \neq C_{i,j}$. This result is by Buhrman and Špalek [1].*

References

[1] H. Buhrman and R. Špalek. Quantum verification of matrix products. In *Proceedings of 17th ACM-SIAM SODA*, pages 880–889, 2006. quant-ph/0409035.

[2] C. Dürr, M. Heiligman, P. Høyer, and M. Mhalla. Quantum query complexity of some graph problems. *SIAM Journal on Computing*, 35(6):1310–1328, 2006. Earlier version in ICALP'04.

[3] R. Freivalds. Probabilistic machines can use less running time. In *IFIP Congress*, pages 839–842, 1977.