Solutions to Exercise Set 12
Quantum Computation

Exercise 1 (From a decision version of factoring to actual factoring)

Complexity classes such as BQP are usually defined for decision problems, whose output is one bit. Factoring, however, is usually presented as a search problem: given an integer N, find its prime factors. This exercise explains why this distinction is not a serious issue for factoring.

Consider the following decision problem:

$$\text{LargePrimeFactor} = \{\langle N, k\rangle : N \text{ has a prime factor } p \text{ with } k \leq p \leq N-1\}.$$

Inputs are written in binary, and $n = \lceil \log_2 N \rceil$ denotes the input length of N up to constant factors. Assume you are given an algorithm A that solves LargePrimeFactor in time $\text{poly}(n)$. Give an algorithm that takes an integer N as input, and uses calls to A in order to return a complete factorization of N. What is the maximum number of calls to A that your algorithm will make, as a function of n?

Solution. The key observation is that, for a fixed integer M, the predicate

$$P_M(k) := A(M, k)$$

is monotone in k. Indeed, if L denotes the largest proper prime factor of M, then

$$P_M(k) = 1 \quad \Longleftrightarrow \quad k \leq L,$$

provided M is composite. If M is prime, then $P_M(k) = 0$ for every $k \geq 2$, because the only prime factor of M is M itself and the definition only allows factors at most $M-1$.

We can therefore find the largest prime factor of a composite integer by binary search. The full algorithm is the following.

1. Start with an empty list of prime factors and set $M = N$.
2. If $M = 1$, stop.
3. Query $A(M, 2)$. If the answer is no, then M has no proper prime factor, hence M is prime. Append M to the list and stop.
4. If the answer is yes, then M is composite. Use binary search on the interval $[2, M-1]$ with the monotone predicate $P_M(k)$ to find the largest integer L such that $P_M(L) = 1$. Then L is the largest prime factor of M.
5. Divide out all powers of L: find the largest $e \geq 1$ such that L^e divides M, append L exactly e times to the list, replace M by M/L^e, and return to step 2.

Let us check correctness. If $A(M, 2)$ says no, then there is no prime factor p of M with $2 \le p \le M-1$, so M is prime. If $A(M, 2)$ says yes, the set of values of k for which $A(M, k)$ says yes is exactly

$$\{2, 3, \ldots, L\},$$

where L is the largest proper prime factor of M. Binary search therefore recovers L. Since L is a prime factor, dividing out all copies of L is legitimate. Repeating this procedure eventually removes all prime factors of N.

Now count oracle calls. For a number $M \le N$, binary search uses at most

$$\lceil \log_2 M \rceil \le n$$

queries, up to an irrelevant additive constant. Each successful binary search discovers at least one new prime factor, and after dividing out all copies of that prime, the remaining integer decreases by a factor of at least 2. Hence this can happen at most n times. There may also be one final query that discovers that the last remaining factor is prime.

Thus the total number of calls to A is at most $O(n^2)$; more explicitly, one can bound it by $n(n+1)+1$ with the above implementation. Since each call to A is polynomial time and the extra arithmetic is also polynomial time in n, the complete factorization is found in polynomial time. ■

Exercise 2 (Local gates and path-sums)

The proof that BQP $\subseteq$ PSPACE is based on a simple idea: one can compute a single amplitude of a quantum circuit by summing over all possible computational paths. This is usually much too slow, but it can be done while reusing memory.

Let S be the number of qubits, and identify basis states with bit strings in $\{0,1\}^S$. For a unitary U, write

$$U_{i,j} = \langle i | \, U \, | j \rangle , \qquad i, j \in \{0,1\}^S.$$

1. Let U apply a Hadamard gate to qubit k and the identity to all other qubits. Give an efficient rule for computing $U_{i,j}$ from the two strings i, j.

2. Let U apply a CNOT with control qubit k and target qubit ℓ, and the identity elsewhere. Give an efficient rule for computing $U_{i,j}$.

3. Repeat part (b) for a Toffoli gate with control qubits k, ℓ and target qubit r. Again, specify exactly when $U_{i,j} = 1$.

Solution. Write

$$i = (i_1, \ldots, i_S), \qquad j = (j_1, \ldots, j_S),$$

with all bits in $\{0,1\}$. The important point is that a local gate can only change the bits on which it acts. Therefore most matrix entries are zero and the nonzero entries can be computed by looking at only a constant number of bits.

1. For a Hadamard on qubit k, all qubits except k must agree. Thus

$$U_{i,j} = 0 \quad \text{unless} \quad i_t = j_t \text{ for every } t \neq k.$$

If this condition holds, then the matrix entry is the corresponding one-qubit Hadamard entry:

$$U_{i,j} = \frac{(-1)^{i_k j_k}}{\sqrt{2}}.$$

This formula gives $1/\sqrt{2}$ for the entries $(0,0),(0,1),(1,0)$ and $-1/\sqrt{2}$ for the entry $(1,1)$.

2. A CNOT with control k and target ℓ maps the input basis state $|j\rangle$ to the basis state $|i\rangle$ defined by

$$i_k = j_k, \qquad i_\ell = j_\ell \oplus j_k, \qquad i_t = j_t \quad \text{for } t \notin \{k, \ell\}.$$

Hence

$$U_{i,j} = \begin{cases} 1, & \text{if the above equalities hold,} \\ 0, & \text{otherwise.} \end{cases}$$

Equivalently, $U_{i,j} = 1$ exactly when i is obtained from j by flipping the target bit j_ℓ if and only if the control bit j_k is 1.

3. A Toffoli gate with controls k, ℓ and target r maps $|j\rangle$ to $|i\rangle$ by keeping all non-target bits fixed and replacing the target bit by

$$i_r = j_r \oplus (j_k j_\ell).$$

Therefore

$$U_{i,j} = \begin{cases} 1, & \text{if } i_t = j_t \text{ for every } t \neq r \text{ and } i_r = j_r \oplus (j_k j_\ell), \\ 0, & \text{otherwise.} \end{cases}$$

In words, the target is flipped exactly when both control bits are equal to 1.

■

Exercise 3 (Using a BQP algorithm as a subroutine)

A classical polynomial-time algorithm can use another classical polynomial-time algorithm as a subroutine. For BQP *this is slightly more delicate, because a bounded-error quantum algorithm does not implement an exact oracle. This exercise shows why the difficulty can be overcome.*

Let $L \in$ BQP, and let $f : \{0,1\}^n \to \{0,1\}$ be its characteristic function: $f(x) = 1$ iff $x \in L$. Assume, after error reduction, that there is a polynomial-size quantum circuit U with the following property. It keeps the input register x unchanged and maps

$$|x\rangle\,|0\rangle\,|0^w\rangle \longmapsto \sqrt{p_x}\,|x\rangle\,|f(x)\rangle\,|\phi_x\rangle + \sqrt{1-p_x}\,|x\rangle\,|1-f(x)\rangle\,|\psi_x\rangle\,,$$

where $w \leq \mathrm{poly}(n)$ and $p_x \geq 1 - \varepsilon$ for every x. Here the middle qubit is the output qubit and the last register is the workspace.

1. *Explain why we may assume the error ε is exponentially small, for example $\varepsilon \leq 2^{-10n}$, while keeping the circuit size polynomial.*

2. *We would like to approximate the ideal bit oracle*

$$O_f : |x\rangle\,|b\rangle \longmapsto |x\rangle\,|b \oplus f(x)\rangle\,.$$

Construct a circuit V using U, $U^\dagger$, and one CNOT gate, such that on inputs of the form

$$|x\rangle\,|b\rangle\,|0\rangle\,|0^w\rangle$$

it approximately implements

$$|x\rangle\,|b\rangle\,|0\rangle\,|0^w\rangle \longmapsto |x\rangle\,|b \oplus f(x)\rangle\,|0\rangle\,|0^w\rangle\,.$$

3. *Show that for each basis input $|x\rangle\,|b\rangle\,|0\rangle\,|0^w\rangle$, the state produced by V has distance $O(\sqrt{\varepsilon})$ from the ideal state.*

4. *The previous part bounds the error on each basis state. Explain why choosing ε exponentially small makes the operator-norm error of V on the whole subspace with clean workspace exponentially small.*

5. *Suppose a polynomial-size quantum circuit for another language L' makes $q(n)$ queries to the ideal oracle O_f, where $q(n) \le \mathrm{poly}(n)$. Replace each query by the approximate circuit V. Use a hybrid argument or triangle inequality to show that the total error introduced is still negligible.*

6. *Conclude that a* BQP *computation may use a* BQP *language as a subroutine without leaving* BQP*. In other words,* BQP *is closed under polynomially many* BQP *subroutine calls.*

Solution.

1. The usual error-reduction argument for BQP applies. Run the original bounded-error circuit independently r times, compute the majority of the r output bits reversibly, and use this majority bit as the new output. By a Chernoff bound, the error decreases like $e^{-\Omega(r)}$. Taking $r = \Omega(n)$ gives error at most 2^{-10n}, and the circuit size only increases by the polynomial factor $O(n)$. The input register can be kept unchanged throughout, and the extra registers are simply included in the workspace.

2. Let the registers be ordered as

$$|x\rangle\,|b\rangle\,|c\rangle\,|z\rangle\,,$$

where b is the target bit of the oracle, c is the output qubit used by U, and z is the workspace of U. Define V as follows:

$$V = U^\dagger\,\mathrm{CNOT}_{c\to b}\,U,$$

where U and $U^\dagger$ act on the registers x, c, z, and the CNOT has control c and target b.

Starting from $|x\rangle\,|b\rangle\,|0\rangle\,|0^w\rangle$, applying U gives

$$\sqrt{p_x}\,|x\rangle\,|b\rangle\,|f(x)\rangle\,|\phi_x\rangle + \sqrt{1-p_x}\,|x\rangle\,|b\rangle\,|1-f(x)\rangle\,|\psi_x\rangle\,.$$

The CNOT from c to b gives

$$\sqrt{p_x}\,|x\rangle\,|b \oplus f(x)\rangle\,|f(x)\rangle\,|\phi_x\rangle + \sqrt{1-p_x}\,|x\rangle\,|b \oplus (1-f(x))\rangle\,|1-f(x)\rangle\,|\psi_x\rangle\,.$$

Finally we apply $U^\dagger$ to x, c, z and this has the desired result.

3. We compare the state just before applying $U^\dagger$ with an ideal state just before uncomputing. Define

$$|\chi_x\rangle = U\,|x\rangle\,|0\rangle\,|0^w\rangle = \sqrt{p_x}\,|x\rangle\,|f(x)\rangle\,|\phi_x\rangle + \sqrt{1-p_x}\,|x\rangle\,|1-f(x)\rangle\,|\psi_x\rangle\,.$$

The ideal state before uncomputation would be

$$|x\rangle\,|b\oplus f(x)\rangle \otimes \Big(\sqrt{p_x}\,|f(x)\rangle\,|\phi_x\rangle + \sqrt{1-p_x}\,|1-f(x)\rangle\,|\psi_x\rangle\Big),$$

which is the same as flipping b by the correct value $f(x)$ while leaving the computed state untouched.

The actual state differs from this ideal pre-uncomputation state only on the bad branch, which has norm $\sqrt{1-p_x} \le \sqrt{\varepsilon}$. More explicitly, the difference vector is

$$\sqrt{1-p_x}\,(|b\oplus(1-f(x))\rangle - |b\oplus f(x)\rangle)\,|x\rangle\,|1-f(x)\rangle\,|\psi_x\rangle\,,$$

so its norm is at most

$$\sqrt{2(1-p_x)} \le \sqrt{2\varepsilon}.$$

Applying $U^\dagger$ cannot change distances. Moreover,

$$U^\dagger\,|\chi_x\rangle = |x\rangle\,|0\rangle\,|0^w\rangle\,.$$

Hence the final state produced by V is within distance $\sqrt{2\varepsilon} = O(\sqrt{\varepsilon})$ of

$$|x\rangle\,|b\oplus f(x)\rangle\,|0\rangle\,|0^w\rangle\,.$$

4. Let E be the difference between the actual operation V and the ideal clean oracle operation, restricted to the subspace spanned by

$$|x\rangle\,|b\rangle\,|0\rangle\,|0^w\rangle\,, \qquad x\in\{0,1\}^n, \quad b\in\{0,1\}.$$

Part (c) says that, on each basis vector of this subspace,

$$\|E\,|x\rangle\,|b\rangle\,|0\rangle\,|0^w\rangle\,\| \le \delta, \qquad \delta := \sqrt{2\varepsilon}.$$

The subspace has dimension 2^{n+1}. Therefore, for any normalized superposition

$$|\alpha\rangle = \sum_{x,b} \alpha_{x,b}\,|x\rangle\,|b\rangle\,|0\rangle\,|0^w\rangle\,,$$

we have

$$\|E\,|\alpha\rangle\,\| \le \sum_{x,b} |\alpha_{x,b}|\,\delta \le \delta\sqrt{2^{n+1}}.$$

Thus the operator-norm error on the clean subspace is at most

$$\sqrt{2\varepsilon}\,2^{(n+1)/2}.$$

If, for example, $\varepsilon \le 2^{-10n}$, then this is exponentially small in n. Taking the error reduction slightly stronger if needed gives any desired inverse-exponential bound.

5. Let η be the operator-norm error of one replacement of O_f by V on the relevant clean subspace. We compare the ideal oracle circuit with the circuit in which all $q(n)$ oracle calls have been replaced by V.

   Introduce hybrid circuits $H_0, H_1, \ldots, H_q$, where H_t uses V for the first t oracle calls and the ideal oracle O_f for the remaining $q-t$ calls. Consecutive hybrids differ in only one query. Since all other gates are unitary, replacing that one query changes the final state by at most η. Therefore

$$\|H_q |\text{in}\rangle - H_0 |\text{in}\rangle\| \leq \sum_{t=1}^{q} \|H_t |\text{in}\rangle - H_{t-1} |\text{in}\rangle\| \leq q(n)\eta.$$

   Since $q(n)$ is polynomial and η is exponentially small, the total error is still negligible. In particular, it can be made much smaller than the constant completeness-soundness gap of a BQP computation.

   A small technical point is worth making explicit. For each simulated oracle call, we use fresh auxiliary registers c, z initialized to $|0\rangle |0^w\rangle$. Thus every invocation of V starts with clean workspace. The operator-norm bound from part (d) also applies when the query register is entangled with the rest of the computation, because we may tensor the bound with the identity on all other registers.

6. From the previous questions it follows that a polynomial-size quantum circuit may call a BQP subroutine polynomially many times as follows: amplify the subroutine so that its error is exponentially small, turn it into a coherent approximate oracle using compute-copy-uncompute, and replace each ideal oracle query by this approximate oracle. The accumulated error remains negligible, so after the usual final amplification the resulting ordinary quantum circuit still decides the language with bounded error. Hence polynomially many BQP subroutine calls do not take us outside BQP.

■

Exercise 4 (A stronger upper bound, BQP $\subseteq$ PP)

This exercise proves a stronger classical upper bound than BQP $\subseteq$ PSPACE. The class PP consists of decision problems solvable by a polynomial-time randomized classical algorithm whose acceptance probability is strictly larger than $1/2$ on yes-instances and strictly smaller than $1/2$ on no-instances. Unlike BPP, the gap above or below $1/2$ may be exponentially small.

For this exercise, assume that the BQP computation is given by a polynomial-size circuit C over Hadamard, Toffoli, and Z gates, acting on $S = \text{poly}(n)$ qubits. The input is $\left|x, 0^{S-n}\right\rangle$. Measuring the first qubit at the end gives $f(x)$ with probability at least $2/3$. You may take for granted that restricting to such a gate set does not change BQP, up to efficient approximation.

1. Let

$$|\theta_x\rangle = C \left|x, 0^{S-n}\right\rangle$$

   and define

$$a_x = \left\langle x, 0^{S-n}\right| C^\dagger Z_1 C \left|x, 0^{S-n}\right\rangle,$$

where Z_1 applies a Z gate to the first qubit. Show that

$$a_x = 1 - 2p_{\text{acc}}(x),$$

where $p_{\text{acc}}(x)$ is the probability that the first qubit is measured as 1. Conclude that

$$f(x) = 0 \Rightarrow a_x \geq \frac{1}{3}, \qquad f(x) = 1 \Rightarrow a_x \leq -\frac{1}{3}.$$

2. Apply the path-sum formula to the circuit $D = C^\dagger Z_1 C$ and the amplitude a_x. Explain why every nonzero computational path contributes either $+2^{-h/2}$ or $-2^{-h/2}$, where h is the total number of Hadamard gates in D. All zero paths contribute 0.

3. Let N_+ be the number of paths contributing $+2^{-h/2}$ and N_- the number of paths contributing $-2^{-h/2}$. Show that the sign of a_x is the sign of $N_+ - N_-$.

4. Design a polynomial-time randomized classical algorithm whose acceptance probability is larger than $1/2$ exactly when $N_- > N_+$. Conclude that $\mathsf{BQP} \subseteq \mathsf{PP}$.

Solution. Let

$$y_0 = (x, 0^{S-n})$$

be the initial computational-basis string.

1. Since $|\theta_x\rangle = C\,|y_0\rangle$, we have

$$a_x = \langle\theta_x|\, Z_1\, |\theta_x\rangle\,.$$

The operator Z_1 has eigenvalue $+1$ on basis states whose first bit is 0, and eigenvalue -1 on basis states whose first bit is 1. Therefore

$$a_x = \Pr[\text{first qubit is 0}] - \Pr[\text{first qubit is 1}] = 1 - 2p_{\text{acc}}(x).$$

If $f(x) = 0$, then the computation outputs 0 with probability at least $2/3$, so

$$p_{\text{acc}}(x) \leq \frac{1}{3}, \qquad a_x = 1 - 2p_{\text{acc}}(x) \geq \frac{1}{3}.$$

If $f(x) = 1$, then $p_{\text{acc}}(x) \geq 2/3$, so

$$a_x = 1 - 2p_{\text{acc}}(x) \leq -\frac{1}{3}.$$

2. Write

$$D = C^\dagger Z_1 C = G_T G_{T-1} \cdots G_1,$$

and insert the identity

$$I = \sum_{z \in \{0,1\}^S} |z\rangle\,\langle z|$$

between consecutive gates. Then

$$a_x = \langle y_0|\, D\, |y_0\rangle = \sum_{y_1,\ldots,y_{T-1} \in \{0,1\}^S} \langle y_0|\, G_T\, |y_{T-1}\rangle\,\langle y_{T-1}|\, G_{T-1}\, |y_{T-2}\rangle \cdots \langle y_1|\, G_1\, |y_0\rangle\,.$$

A choice of intermediate strings $(y_1, \ldots, y_{T-1})$ is a computational path.

Now examine the possible local matrix entries. A Toffoli gate is deterministic on computational-basis states, so each of its entries is either 0 or 1. A Z gate is diagonal with entries $+1$ and -1, and all off-diagonal entries are 0. A Hadamard gate has entries

$$\pm\frac{1}{\sqrt{2}}.$$

Hence, if a path is nonzero, every Toffoli and Z factor contributes magnitude 1, and every Hadamard factor contributes magnitude $1/\sqrt{2}$. If h is the total number of Hadamard gates in D, each nonzero path contributes exactly

$$+2^{-h/2} \quad \text{or} \quad -2^{-h/2}.$$

All inconsistent paths contain at least one zero transition and contribute 0.

3. Therefore the whole sum has the form

$$a_x = N_+ 2^{-h/2} - N_- 2^{-h/2} = (N_+ - N_-)2^{-h/2}.$$

Since $2^{-h/2} > 0$, the sign of a_x is exactly the sign of $N_+ - N_-$. From part (a), this sign is positive when $f(x) = 0$ and negative when $f(x) = 1$.

4. Let M be the total number of computational paths in the above sum. For example, if we choose all intermediate strings independently, then

$$M = 2^{S(T-1)},$$

which can be sampled using $S(T-1) = \text{poly}(n)$ random bits.

The randomized classical algorithm is:

(a) Choose a path uniformly at random.

(b) Compute its contribution by multiplying the relevant local gate entries. This can be done in polynomial time because each gate is local.

(c) If the contribution is negative, accept.

(d) If the contribution is positive, reject.

(e) If the contribution is zero, accept with probability 1/2 and reject with probability 1/2.

Let N_0 be the number of zero paths. The acceptance probability of this algorithm is

$$\frac{N_-}{M} + \frac{1}{2}\frac{N_0}{M} = \frac{1}{2} + \frac{N_- - N_+}{2M},$$

because $M = N_+ + N_- + N_0$. Thus the acceptance probability is strictly larger than 1/2 exactly when

$$N_- > N_+.$$

By part (c), this is exactly the case $a_x < 0$, which by part (a) is exactly the case $f(x) = 1$.

Hence every language in BQP is decided by a polynomial-time randomized classical algorithm whose acceptance probability is above $1/2$ on yes-instances and below $1/2$ on no-instances. Therefore

$$\mathsf{BQP} \subseteq \mathsf{PP}.$$

■