Exercises for Quantum Information
Sheet 3 — Quantum Circuits

The gates T (*Toffoli gate*) and CNOT (*controlled negation*) are defined as follows:

$$T\colon (x,y,z) \mapsto (x,y,z \oplus (x \wedge y)),\ \ \mathsf{CNOT}\colon (x,y) \mapsto (x, y \oplus x),$$

where x, y, z range over $\{0,1\}$, and $\oplus$ is a binary XOR operator (*exclusive or*). The NAND gate (*not and*) is defined as $\mathsf{NAND}\colon (x,y) \mapsto \neg(x \wedge y)$.

Exercise 1. Show that NAND is complete, i.e. any boolean function from $\{0,1\}^n \to \{0,1\}$ can be expressed as a suitable combination of NAND gates. Show that the Toffoli gate is reversible (i.e. bijective), and is complete (i.e., for any boolean function $f\colon \{0,1\}^n \to \{0,1\}$, there is a composition of T gates which takes $x_0, \ldots, x_{n-1}$ with some extra constant bits and outputs $x_0, \ldots, x_{n-1}, f(x_0, \ldots, x_{n-1})$ with some extra constant bits).

Solution. For NAND, it is enough to express NOT, AND, and OR using only NAND:

$$\neg x = x \,\mathrm{NAND}\, x, \qquad x \wedge y = (x \,\mathrm{NAND}\, y) \,\mathrm{NAND}\, (x \,\mathrm{NAND}\, y),$$

and by De Morgan,

$$x \vee y = (\neg x) \,\mathrm{NAND}\, (\neg y) = (x \,\mathrm{NAND}\, x) \,\mathrm{NAND}\, (y \,\mathrm{NAND}\, y).$$

Since every Boolean function can be built from NOT, AND, and OR, NAND is complete.

For the Toffoli gate,

$$T(x,y,z) = (x,y,z \oplus xy).$$

This map is reversible because applying it twice gives back the input:

$$z \mapsto z \oplus xy \mapsto (z \oplus xy) \oplus xy = z.$$

So $T^{-1} = T$, hence T is bijective.

To show completeness, note first that fixing $z = 1$ gives

$$T(x,y,1) = (x,y,\neg(xy)),$$

so Toffoli implements NAND on the third bit, with the first two bits unchanged. Since NAND is complete, any Boolean circuit can be built from Toffoli gates plus constant bits.

Equivalently, one can compute any Boolean function $f(x_0, \ldots, x_{n-1})$ reversibly by using extra ancilla bits initialized to constants and storing intermediate values, so that the output contains the original bits together with $f(x_0, \ldots, x_{n-1})$. Hence Toffoli is reversible and complete. □

Exercise 2. Compute matrices representing CNOT and T gates. Show that the matrices are unitary.

Solution. In the computational basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$, CNOT acts by

$$|00\rangle \mapsto |00\rangle, \quad |01\rangle \mapsto |01\rangle, \quad |10\rangle \mapsto |11\rangle, \quad |11\rangle \mapsto |10\rangle,$$

so its matrix is

$$\mathrm{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

For the Toffoli gate T, in the computational basis $\{|000\rangle, |001\rangle, |010\rangle, |011\rangle, |100\rangle, |101\rangle, |110\rangle, |111\rangle\}$, it fixes all basis states except that it swaps $|110\rangle$ and $|111\rangle$. Hence

$$T = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

Both matrices are permutation matrices, so their columns form an orthonormal basis. Therefore

$$\mathrm{CNOT}^\dagger \mathrm{CNOT} = I, \qquad T^\dagger T = I,$$

and both are unitary. □

Exercise 3. (Deutsch-Josza problem) Provide a (possibly informal) argument that it is necessary to make at least $2^{n-1}+1$ queries when the oracle function f takes only classical (i.e. $0, 1$) inputs. This means that the underlying decision problem is not in P. On the other hand, show that this problem is in coNP. Does this resolve the P versus NP conjecture?

Solution. In the Deutsch–Jozsa problem, $f : \{0,1\}^n \to \{0,1\}$ is promised to be either constant or balanced, and we want to decide which.

Classically, in the worst case one must make at least $2^{n-1}+1$ queries. Indeed, after seeing only 2^{n-1} values, it is still possible that all observed values are, say, 0. Then two cases are still consistent with the answers so far: either f is the constant-0 function, or f is balanced and equals 0 on exactly the queried 2^{n-1} inputs and 1 on the rest. So with only 2^{n-1} queries one cannot always decide; one more query is needed. Since $2^{n-1}+1$ is exponential in n, this brute-force classical decision procedure is not polynomial-time, so this oracle problem is not in P in the black-box model.

On the other hand, the problem is in coNP. A NO-certificate for the statement "f is constant" is simply a pair of inputs x, y such that $f(x) \neq f(y)$. Such a certificate proves that

f is not constant, and it can be verified in polynomial time using two oracle queries. Thus the language "f is constant" lies in coNP.

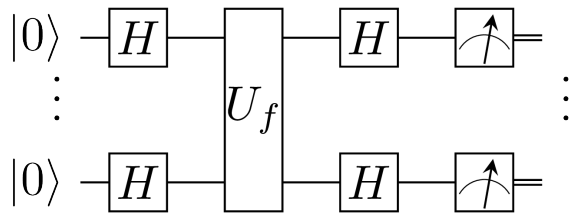
This does not resolve P vs. NP because the argument is in the *oracle (black-box) model*. The lower bound $2^{n-1}+1$ applies only when the function f is accessed via queries, i.e. the algorithm has no description of f other than input-output access.

In contrast, the classes P and NP are defined for problems where the input is given explicitly (e.g. as a string describing a circuit, formula, etc.), and algorithms may exploit the internal structure of the input. Oracle lower bounds do not translate to such settings.

More formally, there are oracle results showing that P vs. NP cannot be resolved by relativizing arguments: there exist oracles A and B such that $\mathrm{P}^A = \mathrm{NP}^A$ and $\mathrm{P}^B \neq \mathrm{NP}^B$. Since the Deutsch–Jozsa lower bound is of this relativizing (oracle) type, it cannot settle P vs. NP in the standard model.

Thus the exponential query complexity here only shows a limitation of classical algorithms in the oracle model, not a separation of P and NP. □

Exercise 4. (Bernstein-Vazirani problem, 1992) Given a boolean function $f\colon :\{0,1\}^n \to \{0,1\}$ which is a dot product between $\overline{x}$ and some fixed string $\overline{s}$, i.e. $f(\overline{x}) = x_1 s_1 \oplus ... \oplus x_n s_n$, compute $\overline{s}$ by querying U_f just once. The quantum circuit solving this problem is depicted below (the ancilla qubit is omitted from the picture). Show that measuring the output of this circuit in



the standard basis yields the desired string $\overline{s}$

Solution. We use the standard phase-oracle form of the Bernstein–Vazirani algorithm. Start with the n input qubits in $|0\rangle^{\otimes n}$ and one ancilla qubit in

$$|-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}.$$

So the initial state is

$$|0\rangle^{\otimes n}|-\rangle.$$

Applying $H^{\otimes n}$ to the first register gives

$$\frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle|-\rangle.$$

Now apply the oracle U_f, defined by

$$U_f|x\rangle|y\rangle = |x\rangle|y \oplus f(x)\rangle.$$

Since the ancilla is $|-\rangle$, this produces phase kickback:

$$U_f|x\rangle|-\rangle = (-1)^{f(x)}|x\rangle|-\rangle.$$

Therefore the state becomes

$$\frac{1}{\sqrt{2^n}}\sum_x (-1)^{f(x)}|x\rangle|-\rangle.$$

To see this, note that we have

$$U_f|x\rangle|-\rangle = \frac{1}{\sqrt{2}}\big(|x\rangle|0\oplus f(x)\rangle - |x\rangle|1\oplus f(x)\rangle\big).$$

If $f(x)=0$, this is unchanged:

$$\frac{1}{\sqrt{2}}(|x\rangle|0\rangle - |x\rangle|1\rangle) = |x\rangle|-\rangle.$$

If $f(x)=1$, the two terms swap:

$$\frac{1}{\sqrt{2}}(|x\rangle|1\rangle - |x\rangle|0\rangle) = -|x\rangle|-\rangle.$$

Thus in both cases

$$U_f|x\rangle|-\rangle = (-1)^{f(x)}|x\rangle|-\rangle.$$

So the value $f(x)$ is not stored in the ancilla, but instead appears as a *phase* $(-1)^{f(x)}$ on $|x\rangle$. This is called phase kickback.

Now use the promise $f(x) = x\cdot s = x_1s_1 \oplus \cdots \oplus x_ns_n$. Then

$$(-1)^{f(x)} = (-1)^{x\cdot s}.$$

So after the oracle the state is

$$\frac{1}{\sqrt{2^n}}\sum_x (-1)^{x\cdot s}|x\rangle|-\rangle.$$

Next apply $H^{\otimes n}$ again to the first register. We have

$$H^{\otimes n}|x\rangle = \frac{1}{\sqrt{2^n}}\sum_{z\in\{0,1\}^n}(-1)^{x\cdot z}|z\rangle.$$

To see this, first recall the single-qubit case:

$$H|0\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle+|1\rangle), \quad H|1\rangle = \tfrac{1}{\sqrt{2}}(|0\rangle-|1\rangle) = \tfrac{1}{\sqrt{2}}\big(|0\rangle+(-1)^1|1\rangle\big).$$

So in general,

$$H|x_i\rangle = \frac{1}{\sqrt{2}}\sum_{z_i\in\{0,1\}}(-1)^{x_iz_i}|z_i\rangle.$$

Now apply $H^{\otimes n}$ to $|x\rangle = |x_1\rangle\cdots|x_n\rangle$:

$$H^{\otimes n}|x\rangle = \bigotimes_{i=1}^n H|x_i\rangle = \bigotimes_{i=1}^n \frac{1}{\sqrt{2}}\sum_{z_i}(-1)^{x_iz_i}|z_i\rangle.$$

Expanding the tensor product gives a sum over all $z = (z_1, \ldots, z_n)$:

$$H^{\otimes n}|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{z\in\{0,1\}^n} \prod_{i=1}^{n} (-1)^{x_i z_i} |z\rangle.$$

Since exponents add,

$$\prod_{i=1}^{n} (-1)^{x_i z_i} = (-1)^{\sum_i x_i z_i} = (-1)^{x\cdot z},$$

where $x \cdot z$ is the bitwise dot product mod 2. Thus

$$H^{\otimes n}|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{z\in\{0,1\}^n} (-1)^{x\cdot z} |z\rangle.$$

We get

$$\frac{1}{2^n} \sum_x \sum_z (-1)^{x\cdot s} (-1)^{x\cdot z} |z\rangle|-\rangle = \frac{1}{2^n} \sum_z \left(\sum_x (-1)^{x\cdot(s\oplus z)} \right) |z\rangle|-\rangle.$$

Now evaluate the inner sum. If $z = s$, then $s \oplus z = 0$, so every term is 1 and

$$\sum_x (-1)^{x\cdot 0} = 2^n.$$

If $z \neq s$, then $s \oplus z \neq 0$, so there is at least one bit where it is 1, and exactly half the strings x give phase $+1$ and half give phase -1. Hence

$$\sum_x (-1)^{x\cdot(s\oplus z)} = 0.$$

Therefore only the term $z = s$ survives, and the final state is

$$|s\rangle|-\rangle.$$

So measuring the first register in the computational basis yields the string s with probability 1. □

Exercise 5. Assume you are given a function $f\colon \{0,1\}^n \to \{0,1\}$ with a quantum oracle gate U_f defined as:

$$U_f : (\overline{x}, y) \mapsto (\overline{x}, y \oplus f(\overline{x})).$$

Let I_0 be the set of n-bit strings with the first bit equal to 0, and I_1 be $\{0,1\}^n \setminus I_0$. You are given a promise that f fulfills one of the two conditions

1. $f(\overline{a}) = 0$ if and only if $\overline{a} \in I_0$;
2. the total number of strings from I_0 for which f is 1 plus the total number of strings from I_1 for which f is 0 is exactly 2^{n-1}.

Design an algorithm that decides which of the two above conditions is fulfilled via just a single query to U_f.

Solution. Let $x = (x_1, \ldots, x_n) \in \{0,1\}^n$, and define

$$b(x) := x_1.$$

Since I_0 is the set of strings whose first bit is 0, and $I_1 = \{0,1\}^n \setminus I_0$, the first promised condition says exactly that

$$f(x) = 0 \iff x \in I_0,$$

which is equivalent to

$$f(x) = x_1 = b(x) \qquad \text{for all } x \in \{0,1\}^n.$$

Now define a new Boolean function

$$g(x) := f(x) \oplus x_1.$$

Then:

- under condition 1, we have $g(x) = 0$ for all x, so g is constant;
- under condition 2, the number of strings $x \in I_0$ with $f(x) = 1$, plus the number of strings $x \in I_1$ with $f(x) = 0$, is exactly 2^{n-1}. But this is precisely the number of inputs x for which $f(x) \neq x_1$, i.e. for which $g(x) = 1$. Hence g is balanced.

Thus the problem is reduced to distinguishing whether g is constant 0 or balanced, which is exactly the Deutsch–Jozsa problem.

We now show how to implement the phase oracle for g using only one query to U_f.

Start in the state

$$|0^n\rangle|1\rangle.$$

Apply $H^{\otimes n}$ to the first register and H to the second register. Since

$$H|1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}} =: |-\rangle,$$

we obtain

$$|0^n\rangle|1\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} |x\rangle|-\rangle.$$

Now apply the oracle U_f. By phase kickback,

$$U_f|x\rangle|-\rangle = (-1)^{f(x)}|x\rangle|-\rangle,$$

so the state becomes

$$\frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)}|x\rangle|-\rangle.$$

Next apply a Z-gate to the first qubit of the first register. Since $Z|x_1\rangle = (-1)^{x_1}|x_1\rangle$, this multiplies each basis state $|x\rangle$ by the phase $(-1)^{x_1}$. Hence the state becomes

$$\frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)}(-1)^{x_1}|x\rangle|-\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)\oplus x_1}|x\rangle|-\rangle = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{g(x)}|x\rangle|-\rangle.$$

Finally apply $H^{\otimes n}$ to the first register. The amplitude of $|0^n\rangle$ is

$$\frac{1}{2^n} \sum_{x\in\{0,1\}^n} (-1)^{g(x)}.$$

Now distinguish the two cases:

1. If condition 1 holds, then $g(x) = 0$ for all x, so

$$\frac{1}{2^n}\sum_x (-1)^{g(x)} = \frac{1}{2^n}\sum_x 1 = 1.$$

   Therefore the first register is exactly $|0^n\rangle$.

2. If condition 2 holds, then g is balanced, so exactly half of the inputs satisfy $g(x) = 0$ and half satisfy $g(x) = 1$. Hence

$$\sum_x (-1)^{g(x)} = 0,$$

   so the amplitude of $|0^n\rangle$ is 0.

Therefore, after measuring the first register:

- if the outcome is 0^n, output that condition 1 holds;
- otherwise, output that condition 2 holds.

This algorithm is exact and uses only a single query to U_f. □