Exercise Set 3
Quantum Computation

Exercise 1 *Deutsch's algorithm*

This algorithm was first imagined by David Deutsch (and then extended together with Richard Josza, to become the algorithm that was presented in the lectures).

The problem is the following : consider a simple Boolean function $f : \{0,1\} \to \{0,1\}$; how many evaluations of this function f are needed in order to decide whether $f(0) = f(1)$ or $f(0) \neq f(1)$? Classically, it is clear that exactly two evaluations of the function f, namely the evaluations of both $f(0)$ and $f(1)$, are needed in order to answer this questions. Deutsch imagined a quantum algorithm that could answer the question with a single evaluation of the function f, more precisely a single call to the quantum oracle U_f with 2-qubits input and 2-qubits output, defined as follows :

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

(a) Construct the quantum circuit for the oracle U_f, for each of the 4 possible functions $f : \{0,1\} \to \{0,1\}$.

(b) Reconstruct then the complete quantum circuit seen in the lectures in this particular case and compute the output of the circuit, explaining how the final measurement allows to decide between the above two alternatives ($f(0) = f(1)$ or $f(0) \neq f(1)$).

(c) *(Coding Exercise)* Implement the Deutsch-Josza algorithm seen in class using the Jupyter notebook "Worksheet 3" (available on Moodle) in qBraid.

Notes :

1. For part (b), *please do not copy-paste* the lectures with the same notations ; rather, redo the exercise from scratch in this particular case !
2. The "quantum advantage" obtained here is mostly theoretical, as it assumes that the oracle U_f is given to us. But building this oracle requires knowing the function f...

Exercise 2 *Bernstein-Vazirani's algorithm*

Consider a vector $a = (a_1, \ldots, a_n) \in \{0,1\}^n$, as well as the function $f : \{0,1\}^n \to \{0,1\}$ defined as

$$f(x) = a \cdot x \,(\text{mod } 2) = a_1x_1 + \cdots + a_nx_n \,(\text{mod } 2) = a_1x_1 \oplus \cdots \oplus a_nx_n, \quad \text{for } x \in \{0,1\}^n$$

Classically, n evaluations of the function f are needed in order to discover the value of vector $a \in \{0,1\}^n$, by considering successively $x^{(1)} = (1,0,\ldots,0)$, $x^{(2)} = (0,1,0,\ldots,0)$, etc.

(a) Show that assuming again the existence of a quantum oracle U_f with $n+1$-qubits input and output, as defined in the lecture on the Deutsch-Josza algorithm, it is possible to discover the value of the vector a with probability 1 and a single call to the oracle U_f.

(b) Consider now a slightly different function f, defined as :

$$f(x) = b \oplus a \cdot x, \quad \text{for } x \in \{0,1\}^n$$

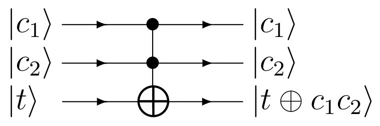
where $b \in \{0,1\}$ is also unknown a priori.

(b.i) With the same circuit as above, is it still possible to discover the value of the vector a with probability 1 and a single call to the oracle U_f ?

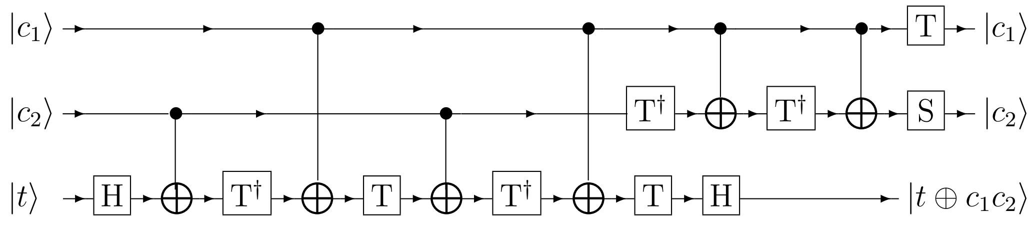
(b.ii) And what about the value of b ? Is it possible to determine this one ?

Exercise 3 *Construction of the Toffoli gate with C-NOT, H, T and S gates*

Verify that the control-control-NOT also called Toffoli gate :



is equivalent to the following circuit made of $CNOT$, H, T and S

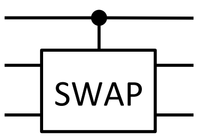


Hints : - Observe first that $(CNOT)\,|x\rangle \otimes |y\rangle = |x\rangle \otimes |x \oplus y\rangle$ can also be represented as $|x\rangle \otimes X^x\,|y\rangle$, where $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ is the NOT gate.

- Then it is also useful to compute first independently $XT^\dagger X$.

Exercise 4 *Fredkin gate*

The SWAP operation takes two input bits and permutes them : SWAP$|b_1, b_2\rangle = |b_2, b_1\rangle$. The Fredkin gate is a three input controlled SWAP gate and is reversible. The gate swaps the two last bits if the first bit is a 1. Otherwise it leaves the input bits unchanged. One intriguing particularity of the Fredkin gate is that it conserves the number of ones.



(a) Show that the irreversible gates AND, OR can be represented in a reversible way from the Fredkin gate.

Hint : Think first how to represent the outputs of the Fredkin gate in the general case.

(b) Give the matrix representation of the Fredkin gate.

(c) Represent the Toffoli (CCNOT) gate in terms of {Fredkin, CNOT}.
Hint : You can achieve this with at most one Fredkin gate and two CNOT gates (and it is helpful to use the matrix representation of these gates).