Quantum Information and Quantum Computing, Solutions 5

Assistant : sara.alvesdossantos@epfl.ch, clemens.giuliani@epfl.ch

Problem 1 : Random number generator

We are going to see how quantum computers can be used to generate genuinely random bit strings.

1. If we want to design a circuit that generates random integers uniformly distributed between 0 and $2^N - 1$, we want every binary string of N bit to be equally probable with probability $\frac{1}{2^N}$.

   This is easy to obtain, by having N qubits in the initial state $|\psi_0\rangle = \bigotimes_N |0\rangle = |0\rangle_N$ and applying a Hadamard gate on every qubit. We obtain

$$\begin{aligned} H^{\otimes N}|0\rangle_N &= \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right)\dots\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) \\ &= \frac{1}{\sqrt{2^N}}\left[|00\dots0\rangle + |00\dots1\rangle + \dots|11\dots1\rangle\right] \\ &= \frac{1}{\sqrt{2^N}}\sum_{x=0}^{2^N-1}|x\rangle_N \end{aligned}$$

   which has exactly the probability amplitudes we were looking for!

2. In this circuit, we want a difference between even and odd integers. For a qubit state (or a generic bit string) being even or odd means having the right-most qubit in $|0\rangle$ or $|1\rangle$, respectively. So, we have to act on the qubit q_0, the one at the top of the circuit.

   We impose that

$$P_{\text{even}} = \frac{2+\sqrt{2}}{2-\sqrt{2}}P_{\text{odd}} \tag{1}$$

   but is also always true that

$$P_{\text{even}} + P_{\text{odd}} = 1 \tag{2}$$

   so we get

$$\begin{aligned} P_{\text{odd}} &= \frac{1}{2} - \frac{1}{2\sqrt{2}} \\ P_{\text{even}} &= \frac{1}{2} + \frac{1}{2\sqrt{2}}\,. \end{aligned}$$

Therefore, for a single odd/even number we get

$$P_{\text{single odd}} = \frac{\sqrt{2}-1}{2^N\sqrt{2}}$$
$$P_{\text{single even}} = \frac{\sqrt{2}+1}{2^N\sqrt{2}}$$

To understand what we have to do, let's consider the simple case of two qubits. We can start with the same circuit seen in the previous point to obtain

$$|\psi_1\rangle = \frac{1}{2}\left[|00\rangle + |01\rangle + |10\rangle + |11\rangle\right] \tag{3}$$

Now, for this state, we obtain $\frac{1}{4}$ as a probability for each state, but we want $P_{\text{single e,o}} = \frac{2\pm\sqrt{2}}{4}$, so we have to modify the coefficient of these states with complex numbers easy to obtain with single-qubits gates and with square modulus $2 \pm \sqrt{2} : 1 \pm e^{i\pi/4}$ (since $|1 \pm e^{i\pi/4}|^2 = 2 \pm \sqrt{2}$), obtained with the T gate

- apply T to q_0 to obtain

$$|\psi_2\rangle = \frac{1}{2}\left[|00\rangle + e^{i\pi/4}|01\rangle + |10\rangle + e^{i\pi/4}|11\rangle\right] \tag{4}$$

- now, with the new phase added, we can apply another H to the first qubit q_0

$$|\psi_3\rangle = \frac{1}{2}\left[(1+e^{i\pi/4})|00\rangle + (1-e^{i\pi/4})|01\rangle + (1+e^{i\pi/4})|10\rangle + (1-e^{i\pi/4})|11\rangle\right] \tag{5}$$

If we now compute the probability to obtain the different states of the computational basis, we observe that they are exactly the probabilities we were looking for.

Since we acted only on the first qubit of the system, this strategy can be easily generalised to the N-qubit case:

$$(HTH) \otimes H^{\otimes N-1}|0\rangle_N \tag{6}$$

is the general scheme for N qubits.

3. Using an arbitrary phase gate instead of the T gate, we have that the states of the computational basis now enter the final state with probability amplitudes $1 \pm e^{i\phi}$. By using

$$(1 \pm \cos\phi)^2 + \sin^2\phi = 2 \pm 2\cos\phi\,, \tag{7}$$

The total probability of obtaining even and odd numbers is respectively

$$P_{\text{even}} = \frac{2+2\cos\phi}{2^N}2^{N-1} = 1+\cos\phi$$
$$P_{\text{odd}} = 1-\cos\phi$$

and their ratio is

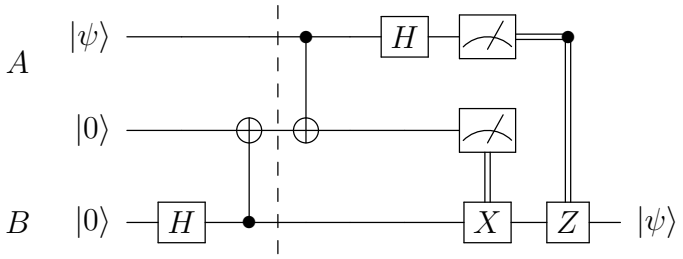
$$x = \frac{1+\cos\phi}{1-\cos\phi} \quad \in \mathbb{R}\,. \tag{8}$$

We can now choose the value of ϕ so to reproduce any probability ratio between even and odd outcomes.

4. In order to generate an arbitrary probability distribution, we would need to generate a state whose components on the computational basis have the corresponding probability amplitudes (while the phase of each component would be irrelevant). Imagine for a moment that we want to generate a state with given amplitudes and given phases. This is obviously a computationally hard problem. It corresponds in fact to the task of generating an arbitrary state, which is known to be computationally hard. Notice that generating an arbitrary state is not the same task as designing a circuit that applies an arbitrary unitary. The latter task is in principle even more complex than the first. However, both are known to be computationally hard in general. Now it turns out that even generating a state where the amplitudes have given values while the phases can be arbitrary is computationally hard. You can learn more on this fact from the literature on *Boson sampling* – one of the first proposals for achieving quantum supremacy which has now lost traction because of fundamental physical limits of the quantum optics hardware needed to realize it experimentally. Also Google's quantum supremacy result of 2019 relies on the generation of an output probability distribution - the Porter-Thomas probability distribution - that is computationally hard to sample with classical computers.

Problem 2 : Quantum Teleportation

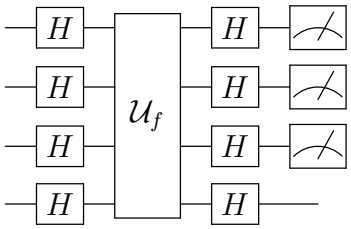
In this exercise we are required to construct a quantum circuit to perform the quantum teleportation protocol on a quantum computer. We know that we need 3 qubits: the qubits that the sender (Alice) wants to transmit plus two entangled qubits shared between the sender and the receiver (Bob). Considered the operations that we have seen in exercise session 2 the corresponding circuit is the following:



(9)

Problem 3 : Oracle for the Deutsch-Josza algorithm

The general circuit to solve a three-qubit Deutsch-Josza problem is the following



(10)

where $\mathcal{U}_f$ is the set of gate that apply the function $f : \{0,1\}^{\times 3} \to \{0,1\}$, taking as an input register the first three qubits and as the output register the fourth.

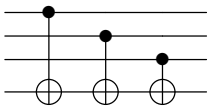
$$\mathcal{U}_f|x\rangle_n|y\rangle_m = |x\rangle_n|y \oplus f(x)\rangle_m \tag{11}$$

The qubits are initialized in the state $|0001\rangle$. In this problem, our aim is to design the $\mathcal{U}_f$ gate in order to apply different functions f

- if $f(x) = 0 \quad \forall x$, we apply no gate
- if $f(x) = 1 \quad \forall x$, we simply apply X gate on the $|1\rangle$ qubit

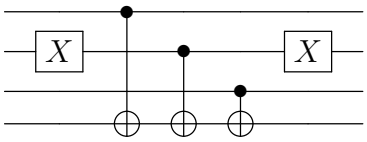
For balanced functions, the situation is more complicated and different for each possible function. In particular, in our case:

- to obtain $f(x) = 1$ for x with an odd number of 1, one way to do it is to act with a CNOT on every qubit of the input register, so that if a odd number of qubits is in $|1\rangle$ the output qubit will be flipped



(12)

- to obtain $f(x) = 1$ for x with an even number of 1, we use the same circuit as before, but act with a X gate at the beginning and at the end of the circuit of a qubit in order to change the number of qubit equal to $|1\rangle$ in the input register and then shifting it back to its original value after the CNOTs are applied.



(13)