Quantum Computing Exercises, September 29, 2000

1. **Simon's algorithm (+ solution)**
   In this exercise we examine a run of Simon's algorithm on the following function f (with $n = 3$):

   $$\begin{aligned} f(000) &= f(111) = 000 \\ f(001) &= f(110) = 001 \\ f(010) &= f(101) = 010 \\ f(011) &= f(100) = 011 \end{aligned}$$

   Note that f is 2-to-1 and $f(x) = f(x \oplus 111)$ for all x, so $s = 111$.

   (a) The starting state of Simon's algorithm: $|000\rangle|000\rangle$

   (b) The state after the first Hadamard transforms on the first 3 qubits:

   $$\frac{1}{\sqrt{8}} \sum_{y \in \{0,1\}^3} |y\rangle|000\rangle$$

   (c) The state after applying f:

   $$\frac{1}{\sqrt{8}}(|000\rangle|000\rangle + |001\rangle|001\rangle + |010\rangle|010\rangle + |011\rangle|011\rangle + |100\rangle|011\rangle + |101\rangle|010\rangle + |110\rangle|001\rangle + |111\rangle|000\rangle)$$

   (d) The state after measuring the second register (supposing this measurement gave $|001\rangle$):

   $$\frac{1}{\sqrt{2}}(|001\rangle + |110\rangle) \otimes |001\rangle$$

   (e) Using that $H^{\otimes n}|x\rangle = \frac{1}{\sqrt{2^n}} \sum_{y \in \{0,1\}^n} (-1)^{x \cdot y}|y\rangle$ we get the state after the final Hadamard transforms:

   $$\begin{aligned} &\frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{8}} \sum_{y \in \{0,1\}^3} (-1)^{001 \cdot y}|y\rangle + \frac{1}{\sqrt{8}} \sum_{y \in \{0,1\}^3} (-1)^{110 \cdot y}|y\rangle\right) \otimes |001\rangle = \\ &\frac{1}{4}\left(\sum_{y \in \{0,1\}^3} ((-1)^{001 \cdot y} + (-1)^{(001 \oplus s) \cdot y})|y\rangle\right) \otimes |001\rangle = \\ &\frac{1}{4}\left(\sum_{y \in \{0,1\}^3} (-1)^{001 \cdot y}(1 + (-1)^{s \cdot y})|y\rangle\right) \otimes |001\rangle = \\ &\frac{1}{2}(|000\rangle - |011\rangle - |101\rangle + |110\rangle) \otimes |001\rangle \end{aligned}$$

   (f) Why does a measurement of the first 3 qubits of this final state give information about s?
   Because the $|y\rangle$ that you measure satisfies $y \cdot s = 0 \mod 2$, which gives information about s. For instance, if the first run of the algorithm gives $y = 011$ then you know $011 \cdot s = 0$, i.e. the second and third bit of s are equal. If the second run gives for instance $y = 101$ then you know the first and the third bit of s are equal. Together with $s \neq 000$ this already gives sufficient information to conclude that $s = 111$. In general, $O(n)$ repetitions of the above experiment will almost always give sufficient information to determine s.

2. **Yao-principle**

Let P be some problem which takes some $f : \{0,1\}^n \to \{0,1\}^m$ as argument and maps it to some r-bit string $P(f) \in \{0,1\}^r$. Let $R(P)$ denote the minimal number of f-queries required by a classical *randomized* algorithm that outputs the right value $P(f)$ with probability at least $2/3$, for all possible inputs f. If μ is a probability distribution on all f, then let $D^\mu(P)$ denote the minimal number of f-queries required by a classical *deterministic* algorithm that outputs the right value $P(f)$ with probability $2/3$ when the input f is drawn according to the distribution μ. The *Yao-principle* says:

$$R(P) = \max_\mu D^\mu(P).$$

In a way, it says that for classical algorithms, randomness inside the algorithm is equivalent to randomness on the inputs. One half of this principle says that for every input distribution μ we have the following lower bound on $R(P)$:

$$R(P) \geq D^\mu(P).$$

Accordingly, to prove a lower bound on *randomized* algorithms it suffices to find a "hard distribution" μ and prove a lower bound on *deterministic* algorithms that give the correct output on most inputs, where "most" is measured according to the distribution μ. In the monday-lecture this principle was used to prove an exponential lower bound on the complexity $R(Simon)$ of Simon's problem via a lower bound on $D^\mu(Simon)$ (with μ the uniform distribution on all f that satisfy the Simon-promise).

To illustrate the Yao-principle, consider the PARITY function. This takes arbitrary $f : \{0,1\}^n \to \{0,1\}$ as argument and gives $\text{PARITY}(f) = 1$ if the number of x with $f(x) = 1$ is odd, and gives $\text{PARITY}(f) = 0$ if that number is even. I.e., $\text{PARITY}(f) = \sum_{x\in\{0,1\}^n} f(x) \mod 2$.

(a) Prove that $R(\text{PARITY}) = 2^n$.
(Hint: use the Yao-principle with uniform distribution $\mu(f) = 1/2^{2^n}$ on all f.)

(b) Give a quantum algorithm that computes $\text{PARITY}(f)$ using only $\frac{1}{2} \cdot 2^n$ f-queries.
(Hint: remember the Deutsch-problem.)