Microsoft Q# Coding Contest - Summer 2018 - Warmup Round
June 29 - July 2, 2018

Mariia Mykhailova and Chris Granade
Quantum Architectures and Computation Group, Microsoft Research, Redmond, WA, United States

EXAMPLE TASKS

A. Generate plus state or minus state

You are given a qubit in state $|0\rangle$ and an integer *sign*.

Your task is to convert the given qubit to state $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ if $sign = 1$ or $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ if $sign = -1$.

You have to implement an operation which takes a qubit and an integer as an input and has no output. The "output" of your solution is the state in which it left the input qubit.

Solution. Look up a list of quantum gates, for example, Quantum logic gate @ Wikipedia. One of the first gates there will be Hadamard - a gate which converts $|0\rangle$ into $|+\rangle$ and $|1\rangle$ into $|-\rangle$. Let's use it to create the required states.

The given qubit starts in $|0\rangle$ state, so for the case of $sign = 1$ it's sufficient to apply a Hadamard gate to it.

For the case of $sign = -1$, you need to convert the state of the given qubit to $|1\rangle$ before applying a Hadamard; this can be done by applying a Pauli X gate.

You can look up the syntax of conditional statements in Q# documentation.

Listing 1. Generate $|+\rangle$ or $|-\rangle$ state

```
namespace Solution {
    open Microsoft.Quantum.Primitive;
    open Microsoft.Quantum.Canon;

    operation Solve (q : Qubit, sign : Int) : ()
    {
        body
        {
            if (sign == -1) {
                X(q);
            }
            H(q);
        }
    }
}
```

D. Distinguish plus state and minus state

You are given a qubit which is guaranteed to be either in $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ or in $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ state.

Your task is to perform necessary operations and measurements to figure out which state it was and to return 1 if it was a $|+\rangle$ state or -1 if it was $|-\rangle$ state. The state of the qubit after the operations does not matter.

You have to implement an operation which takes a qubit as an input and returns an integer.

Solution. For an introduction to the concept of measurement, see The Qubit article. The qubit states that have to be distinguished are orthogonal (their scalar product $\langle +|-\rangle = 0$), and thus can be distinguished with certainity.

The computational basis measurement, implemented by Q# operation `M`, allows you to reliably distinguish states $|0\rangle$ and $|1\rangle$. Thus, you can use this measurement to distinguish $|+\rangle$ and $|-\rangle$ if you can come up with a transformation which maps $|+\rangle$ to $|0\rangle$ and $|-\rangle$ to $|1\rangle$ (or vice versa).

This transformation is the inverse (or, in quantum computing terms, adjoint) of the transformation discussed in problem A, which was done by a Hadamard gate. Since Hadamard gate is self-adjoint (it is its own adjoint), the necessary transformation is also done by a Hadamard gate.

After the original qubit state is transformed to $|0\rangle$ or $|1\rangle$, you can measure it using `M` operation; if the measurement result is `Zero`, the measured state is $|0\rangle$ and thus the original state was $|+\rangle$, otherwise the original state was $|-\rangle$.

Listing 2. Distinguish $|+\rangle$ and $|-\rangle$ states by transforming them into $|0\rangle$ and $|1\rangle$

```
namespace Solution {
    open Microsoft.Quantum.Primitive;
    open Microsoft.Quantum.Canon;

    operation Solve (q : Qubit) : Int
    {
        body
        {
            H(q);
            if (M(q) == Zero) {
                return 1;
            }
            return -1;
        }
    }
}
```

Alternatively, you can use a measurement which is not a computational basis measurement. For details on this, see Pauli measurements article. Note that $|+\rangle$ and $|-\rangle$ states are eigenstates of Pauli X matrix with eigenvalues +1 and -1, respectively. So measuring the given qubit in PauliX basis without any preliminary transformations will give the result: if the measurement result is `Zero`, the eigenvalue is +1 and thus the measured state is $|+\rangle$, otherwise the state is $|-\rangle$.

The operation which performs measurement in arbitrary Pauli basis instead of the default Pauli Z is `Measure`.

Listing 3. Distinguish $|+\rangle$ and $|-\rangle$ states using measurement in Pauli X basis

```
namespace Solution {
    open Microsoft.Quantum.Primitive;
    open Microsoft.Quantum.Canon;

    operation Solve (q : Qubit) : Int
    {
        body
        {
            if (Measure([PauliX], [q]) == Zero) {
                return 1;
            }
            return -1;
        }
    }
}
```

G. Oracle for f(x) = k-th element of x

Implement a quantum oracle on N qubits which implements a function $f(x) = x_k$, i.e. the value of the function is the value of the k-th qubit.

You have to implement an operation which takes the following inputs:

- an array of qubits x (input register),
- a qubit y (output qubit),
- 0-based index of the qubit from input register k ($0 \leq k < \text{Length}(x)$).

The operation doesn't have an output; the "output" of your solution is the state in which it left the qubits.

An introduction to quantum oracles

An oracle O is a "black box" operation that is used as input to another algorithm. Often, such operations are defined using a classical function $f : \{0,1\}^n \to \{0,1\}^m$ which takes n-bit binary input and produces an m-bit binary output. To do so, consider a particular binary input $x = (x_0, x_1, \ldots, x_{n-1})$. We can label qubit states as $|\boldsymbol{x}\rangle = |x_0\rangle \otimes |x_1\rangle \otimes \cdots \otimes |x_{n-1}\rangle$.

We may first attempt to define O so that $O|x\rangle = |f(x)\rangle$, but this has a couple problems. First, f may have a different size of input and output ($n \neq m$), such that applying O would change the number of qubits in the register. Second, even if $n = m$, the function may not be invertible: if $f(x) = f(y)$ for some $x \neq y$, then $O|x\rangle = O|y\rangle$ but $O^\dagger O|x\rangle \neq O^\dagger O|y\rangle$. This means we won't be able to construct the adjoint operation $O^\dagger$, and oracles have to have an adjoint defined for them.

We can deal with both of these problems by introducing a second register of m qubits to hold our answer. Then we will define the effect of the oracle on all computational basis states: for all $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$,

$$O(|x\rangle \otimes |y\rangle) = |x\rangle \otimes |y \oplus f(x)\rangle\,.$$

Now $O = O^\dagger$ by construction, thus we have resolved both of the earlier problems.

Importantly, defining an oracle this way for each computational basis state $|x\rangle|y\rangle$ also defines how O acts for any other state. This follows immediately from the fact that O, like all quantum operations, is linear in the state that it acts on. Consider the Hadamard operation, for instance, which is defined by $H|0\rangle = |+\rangle$ and $H|1\rangle = |-\rangle$. If we wish to know how H acts on $|+\rangle$, we can use that H is linear,

$$\begin{aligned} H|+\rangle &= \frac{1}{\sqrt{2}}H(|0\rangle + |1\rangle) = \frac{1}{\sqrt{2}}(H|0\rangle + H|1\rangle) \\ &= \frac{1}{\sqrt{2}}(|+\rangle + |-\rangle) = \frac{1}{2}(|0\rangle + |1\rangle + |0\rangle - |1\rangle) = |0\rangle\,. \end{aligned}$$

In the case of defining our oracle O, we can similarly use that any state $|\psi\rangle$ on $n+m$ qubits can be written as

$$|\psi\rangle = \sum_{x\in\{0,1\}^n, y\in\{0,1\}^m} \alpha(x,y)\,|x\rangle\,|y\rangle\,,$$

where $\alpha : \{0,1\}^n \times \{0,1\}^m \to \mathbb{C}$ represents the coefficients of the state $|\psi\rangle$. Thus,

$$\begin{aligned} O|\psi\rangle &= O \sum_{x\in\{0,1\}^n, y\in\{0,1\}^m} \alpha(x,y)\,|x\rangle\,|y\rangle \\ &= \sum_{x\in\{0,1\}^n, y\in\{0,1\}^m} \alpha(x,y) O|x\rangle\,|y\rangle \\ &= \sum_{x\in\{0,1\}^n, y\in\{0,1\}^m} \alpha(x,y)\,|x\rangle\,|y \oplus f(x)\rangle\,. \end{aligned}$$

Solution. In this case, we want to define an oracle for $f(x) = x_k$ as an operation O which transforms a state $|x\rangle|y\rangle$ into a state $|x\rangle|y \oplus x_k\rangle$ for any value of x that corresponds to a basis vector. Since this function has a single output bit ($m = 1$), our code has to flip the output register $|y\rangle$ when $x_k = 1$, and to do nothing if $x_k = 0$. That is,

$$\begin{aligned} O|x\rangle|y\rangle &= |x\rangle|y \oplus f(x)\rangle \\ &= |x\rangle|y \oplus x_k\rangle \\ &= |x\rangle \otimes \begin{cases} |y\rangle & \text{if } x_k = 0 \\ |\neg y\rangle & \text{if } x_k = 1 \end{cases}. \end{aligned}$$

Returning to the list of quantum gates, two-qubit operation `CNOT` is the gate which, when applied to k-th qubit of register x as control and qubit y as a target, performs exactly this transformation.

Listing 4. Oracle for $f(x) = x_k$

```
namespace Solution {
    open Microsoft.Quantum.Primitive;
    open Microsoft.Quantum.Canon;

    operation Solve (x : Qubit[], y : Qubit, k : Int) : ()
    {
        body
        {
            CNOT(x[k], y);
        }
    }
}
```