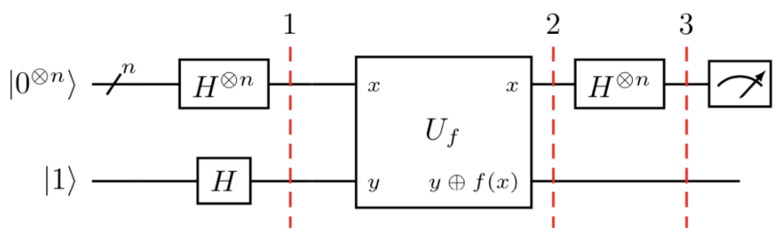
Lecture of April 2nd, 2020

April 1, 2020

1 Deutsch's Algorithm: Setup



We consider the case where $n = 1$. There are only four possible classical functions f of interest:

$$\begin{array}{rcl} \mathit{id}\ 0 & = & 0 \\ \mathit{id}\ 1 & = & 1 \\ \\ \mathit{not}\ 0 & = & 1 \\ \mathit{not}\ 1 & = & 0 \\ \\ c_0\ 0 & = & 0 \\ c_0\ 1 & = & 0 \\ \\ c_1\ 0 & = & 1 \\ c_1\ 1 & = & 1 \end{array}$$

The first two are *balanced* and the last two are *constant*.

The problem we are trying to solve is the following: Given an unknown function f from this collection, determine if it is balanced or constant. Classically this requires two calls to f. The above circuit, as will see, needs just one call to f.

2 The Block U_f

When viewed as a black box, U_f behaves as follows:

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

where $\oplus$ is exclusive-or. Let's expand this for our four functions. We use $\bar{\cdot}$ to denote boolean negation:

$$\begin{aligned}
U_{id}|0,y\rangle &= |0,y\rangle \\
U_{id}|1,y\rangle &= |1,\overline{y}\rangle \\
\\
U_{not}|0,y\rangle &= |x,\overline{y}\rangle \\
U_{not}|1,y\rangle &= |x,y\rangle \\
\\
U_{c_0}|x,y\rangle &= |x,y\rangle \\
\\
U_{c_1}|x,y\rangle &= |x,\overline{y}\rangle
\end{aligned}$$

Like we did last lecture, let's calculate the behavior in the hadamard basis as well. For convenience, we recall how boolean negation behaves in the hadamard basis as well:

$$\begin{aligned}
X|+\rangle &= X(|0\rangle + |1\rangle) = |1\rangle + |0\rangle = |+\rangle \\
X|-\rangle &= X(|0\rangle - |1\rangle) = |1\rangle - |0\rangle = -(|0\rangle - |1\rangle) = -|-\rangle \\
\\
U_{id}|+,+\rangle &= |0,+\rangle + |1,+\rangle = |++\rangle \\
U_{id}|+,-\rangle &= |0,-\rangle - |1,-\rangle = |--\rangle \\
U_{id}|-,+\rangle &= |0,+\rangle - |1,+\rangle = |-+\rangle \\
U_{id}|-,-\rangle &= |0,-\rangle + |1,-\rangle = |+-\rangle \\
\\
U_{not}|+,+\rangle &= |0,+\rangle + |1,+\rangle = |++\rangle \\
U_{not}|+,-\rangle &= -|0,-\rangle + |1,-\rangle = -|--\rangle \\
U_{not}|-,+\rangle &= |0,+\rangle - |1,+\rangle = |-+\rangle \\
U_{not}|-,-\rangle &= -|0,-\rangle - |1,-\rangle = -|+-\rangle \\
\\
U_{c_0}|x,y\rangle &= |x,y\rangle \\
\\
U_{c_1}|x,+\rangle &= |x,1\rangle + |x,0\rangle = |x,+\rangle \\
U_{c_1}|x,-\rangle &= |x,1\rangle - |x,0\rangle = -|x,-\rangle
\end{aligned}$$

So we can trace the execution of Deutsch's algorithm in the four possible cases:

- $f = id$

$$|0,1\rangle \mapsto |+-\rangle \mapsto |--\rangle \mapsto |1-\rangle \mapsto 1$$

- $f = not$

$$|0,1\rangle \mapsto |+-\rangle \mapsto -|--\rangle \mapsto -|1-\rangle \mapsto 1$$

- $f = c_0$

$$|0,1\rangle \mapsto |+-\rangle \mapsto |+-\rangle \mapsto |0-\rangle \mapsto 0$$

- $f = c_1$

$$|0,1\rangle \mapsto |+-\rangle \mapsto -|+-\rangle \mapsto -|0-\rangle \mapsto 0$$

In other words, if we measure 1 the function is balanced and if we measure 0 the function is constant.

3 More Details

`https://qiskit.org/textbook/ch-algorithms/deutsch-josza.html`

4 The Oracle

How do we implement this oracle? And once we implement it, is the number of calls to the oracle a good measure of efficiency?

Oracle for $f = id$

```
circuit = QuantumCircuit(2, 1)
circuit.x(1)
circuit.barrier()

circuit.h(0)
circuit.h(1)

#######
circuit.cx(0,1)
#######

circuit.h(0)

circuit.measure([0],[0])
```

Oracle for $f = not$

```
circuit = QuantumCircuit(2, 1)
circuit.x(1)
circuit.barrier()

circuit.h(0)
circuit.h(1)

#######
circuit.x(0)
circuit.cx(0,1)
circuit.x(0)
#######

circuit.h(0)

circuit.measure([0],[0])
```

Oracle for $f = c_0$

```
circuit = QuantumCircuit(2, 1)
circuit.x(1)
circuit.barrier()

circuit.h(0)
circuit.h(1)

#######
circuit.iden(0)
circuit.iden(1)
#######
```

```
circuit.h(0)

circuit.measure([0],[0])
```

Oracle for $f = c_1$

```
circuit = QuantumCircuit(2, 1)
circuit.x(1)
circuit.barrier()

circuit.h(0)
circuit.h(1)

#######
circuit.iden(0)
circuit.x(1)
#######

circuit.h(0)

circuit.measure([0],[0])
```