My Quantum Model

Korben Rusek

7-22-2019

1 Classical Computation with Quantum Rules

1.1 Classic Computation with Quantum Notation

I assume that readers are familiar with ”classical” computation. Nevertheless, we will begin with a brief introduction to some classical computation. The difference is that we will approach it with the rules of quantum compuation.

At its simplest form classical computation has a collection of bits and a few operations. We work with a collection of n bits, where each bit only has two possible values, 0 or 1. To follow quantum computing notation we will write our n bits as:

$$b = |b_0 b_1 \cdots b_n\rangle \, , b_i \in \{0, 1\}.$$

For example, suppose we have 4 bits and we want to represent the number 2. This would look like $|0010\rangle$. There really is nothing magical here, we are just writing our normal bitstring but with $|\cdot\rangle$ surrounding the bits.

To make things a little easier we will introduce some notation. Let b be a bitstring. Then we will define b_i to be the bit at the i^{th} location. That is, if $b = |0110\rangle$ then $b_0 = |0\rangle$, $b_1 = |1\rangle$, $b_2 = |1\rangle$, and $b_3 = |0\rangle$. Notice that even though b_i is a single bit we can also consider it to be a bitstring.

1.2 X as a NOT gate

Now bits really are not very useful by themselves. We need to introduce some operations to get any utility out of them. In classical computation the simplest of gates is the NOT gate. That is, a gate that simply flips a bit from $|0\rangle$ to $|1\rangle$ or from $|1\rangle$ to $|0\rangle$. To follow quantum computing convention we will call that operator X instead of NOT.

1.3 CNOT and controlled gates

So far as have bits and we have the ability to flip the bits. But as yet, we have no way to make changes to bits that depend on other bits. For that we will introduce the ”controlled” X gate (CX for short). There are several ways to look at CNOT.

2 Borrowing Qubits

2.1 Why AND and OR don't work

2.2 Borrowing Qubits