A Brief Introduction to Quantum Computation

January 15, 2023

1 From Randomness to Qubits

What are Qubits? That's usually the first question getting addressed in any introduction to quantum computing, for a good reason. If we want to construct a new computational model, we first need to define the most basic building block: a single *bit* of information. In classical computer science, the decision on how to define this smallest building block of information seems quite straight forward. We just take the most basic logical fact: either something is *true* or *false*, either 1 or 0. We have a name for an object holding this information: a **Bit**. Let's envision a computational model based on logical gates. Such a gate has one or more inputs and an output, with each either being *true* or *false*. Now consider a bit b and a gate $f : \{0,1\} \to \{0,1\}$. We have a *bit* of information b and can get another *bit* of information $b' \coloneqq f(b)$. In a final third step, we introduce a timescale, which means that now our *bit* of information is time dependent. It can have different values at different times. To make it easier, we choose a discrete timescale. Our Bit b has a distinct value on each point on the timescale. A value of a bit can only be changed in between time steps, by applying a logical gate to it:

$$\begin{array}{cccccccccc} \text{Bit} & b & \xrightarrow{f_1} & b & \xrightarrow{f_2} & \cdots & \to & b & \xrightarrow{f_k} & b \\ \text{time} & t_0 & \to & t_1 & \to & \cdots & \to & t_{k-1} & \to & t_k \end{array}$$

Of course, we need more than one bit of information, if we want to be able to perform meaningful computations. For this, we simply look at a list, vector or register of bits $\mathbf{b} \in \{0,1\}^n$ and modify our gates to be functions $f : \{0,1\}^n \to \{0,1\}^n$ mapping from bit vectors to bit vectors.

Let's recap: We've now designed a computational model with just three components.

- A notion of Information: bits and registers.
- A way of reasoning: logical gates.
- A dimension to do the reasoning in: the timescale

Notice how the system described above is fully deterministic. The state $\mathbf{b}_l$ of our system at time t_l recursively defined by:

$$\mathbf{b}_l = \begin{cases} f_l(\mathbf{b}_{l-1}) & \text{if} \quad l > 0 \\ \mathbf{b}_0 & \text{otherwise} \end{cases}$$

Or by the composition of all gate applications up to this point: $(f_l \circ f_{l-1} \circ \cdots \circ f_1)(\mathbf{b}_0)$. Actually, a composition of gates is also just another logical gate $F \coloneqq (f_l \circ f_{l-1} \circ \cdots \circ f_1) : \{0,1\}^n \to \{0,1\}^n$. If we are not interested in intermediate states, we can thus define our computation in the form of $\mathbf{b}_{\text{out}} \coloneqq F(\mathbf{b}_{\text{in}})$', with '$F : \{0,1\}^n \to \{0,1\}^n$.

1.1 A bit of randomness

Many real world problems are believed (if $\mathbf{P} \neq \mathbf{NP}$) to don't be efficiently solvable on fully deterministic computers like the model described above. Fortunately, it turns out that if we allow for some randomness in our algorithms, we're often able to efficiently find solutions for such hard problems with sufficiently large success probabilities. Often times, the error probabilities can even be made exponentially small. For this reason, we also want to introduce randomness into our model. Algorithms or computational models harnessing the power of randomness are usually called *probabilistic*.

Again, we start with simple one bit systems. Later, we'll see how to expand the following methods to full bit vectors/registers. In the deterministic single bit model above, the state transition of a bit b in step t is defined by $f_t(b) \in \{0,1\}$. Now, the transition function (or gate) is simply allowed to flip an unfair coin and either output 0 or 1 for heads or tails respectively. Of course, the state of b prior to the transition should have an effect on the computation. That is, why we allow different (unfair) coins for either $b = 0$ or $b = 1$. To distinguish between deterministic and probabilistic transition functions, we will denote the latter by $p(b) \in \{0,1\}$. Or to reformulate this idea: Depending on the value of b, the output of $p(b)$ follows one of two Bernoulli trials with success probabilities $0 \leq p_0, p_1 \leq 1$.

$$P(p(b) = 1 \mid b = 0) = p_0$$
$$P(p(b) = 1 \mid b = 1) = p_1$$

Note that we regain our deterministic transition function f from above, if we restrict the probabilities: $p_0, p_1 \in \{0,1\}$. At this point, we can randomize our computation from above as follows:

$$\begin{array}{cccccccccc} \text{Bit} & b & \overset{p_1}{\to} & b & \overset{p_2}{\to} & \cdots & \to & b & \overset{p_k}{\to} & b \\ \text{time} & t_0 & \to & t_1 & \to & \cdots & \to & t_{k-1} & \to & t_k \end{array}$$

Let's have a look at the state of b after the first transition. In the deterministic model, we know with certainty that at this point in time, b will have the value $f_1(b)$. In a probabilistic model, we can not predict the value of b at time t_1 with 100% certainty. In the terminology of probability theory, a probabilistic state transition or even the whole

computation would be an *experiment* and the value of bit b at time t would be described by a *random variable* X_t. Random variables are defined to take a value out of a set of predefined value options $\Omega = \{\omega_1, \ldots, \omega_n\}$ with certain probabilities $p_1, \ldots, p_n$ for each value. Only after we perform the experiment and *observe* its outcome, we get a specific value x_t of the random variable X_t. We say that x_t is a *random sample* or realization of X_t. If we don't want to or can't sample (perform) the experiment, we still could compute the *expected value* $E(X_t) = \sum_i p_i \omega_i$ (if Ω mathematically allows for such operations).

Let's return to our example: The expected state of b at time t_1 is $E(p_1(b) \mid b = 0) = (1 - p_0) \cdot 0 + p_0 \cdot 1$, if we assume that b was in state 0 at time t_0. We say that b is in a (linear) *superposition* of both states 0 and 1.

There are 4 possible transitions with probabilities p_{00}, p_{01}, p_{10} and p_{11}, where p_{ij} is the probability of b transitioning form i to j. Obviously, $\sum_j p_{ij} = 1$ always needs to be satisfied.

1.2 Making it Quantum