

A quick introduction to quantum programming

Jules Jacobs

September 22, 2021

0.1 Quantum states

Imagine that we have a box with some physical system inside of it, with a finite set S of possible states. A probability distribution over S is a vector \vec{p} of probabilities, one probability $p_i \in [0, 1]$ for each state $x \in S$, such that $\sum_x p_x = 1$.

A *quantum state* over S , on the other hand, is a vector $\vec{\phi}$ of *probability amplitudes*, one complex number $\phi_x \in \mathbb{C}$ for each state $x \in S$. If we *measure* such a quantum state, we obtain the outcome x with probability $p_x = |\phi_x|^2$. Thus, in order for ϕ to be a proper quantum state, we must have $\sum_x |\phi_x|^2 = 1$.

0.2 Time evolution

Imagine that the system in the box evolves in time according to some laws of physics. In quantum mechanics, these laws of physics are given by a matrix U that multiplies the state every time step. If the state is currently ϕ , then at the next time step the state is $U\phi$. If there are $n = |S|$ possible states, then U is an $n \times n$ matrix. Only matrices that preserve the condition that the probabilities sum to 1 are allowed: if $\sum_x |\phi_x|^2 = 1$ we must have $\sum_x |(U\phi)_x|^2 = 1$. Such matrices are called *unitary*.

It might be helpful to compare with probabilistic evolution of the state as in a Markov chain. In that case we model the state with a probability vector \vec{p} and we multiply this vector with a matrix M at each time step. If the state is currently p , then at the next time step the state is Mp . Matrices that preserve the condition that all probabilities are non-negative and that their sum remains 1 are called *stochastic matrices*. The entry M_{xy} of the matrix is the probability that the system will step to state y , if the state is currently x . Similarly, the entry U_{xy} of the unitary matrix, is the probability amplitude of next state being y , if the state is currently x .

0.3 What a quantum computer is

A quantum computer with state set S is a device where we can *input* such a matrix U and an initial state ϕ . It will then do one step of time evolution to $\phi' = U\phi$, and it will *measure* the new state ϕ' and tell us which outcome $x \in S$ it got. This outcome is random, and we will get answer x with probability $|(U\phi)_x|^2$. Thus, a quantum computer is a kind of universal quantum mechanics simulator, where we *input* the laws of physics U .

0.4 Quantum circuits

In physics, the state set S is often infinite, and sometimes even uncountably infinite (e.g. the position of a particle), but in quantum programming the set S is taken to be strings of k bits, so that $|S| = 2^k$. The matrix U is a 2^k -by- 2^k matrix. One might wonder how we even input the U to the quantum computer, if it contains an exponential amount of data.

The answer is that we can't quite input *any* matrix U ; it must be encoded as a *quantum circuit*. A quantum circuit is a list of operations we do on the state x of n bits, where each operation operates on some small subset of the bits and leaves the rest of the bits alone.

Often, a small set of operations is used, such as the *Hadamard gate* and the *CNOT gate*. The Hadamard gate operates one bit, and the CNOT gate operates on two bits.

In order to describe what they do, we have to introduce a bit of notation. We use the notation $\phi = |01001\rangle$ for the state ϕ where $\phi_{01001} = 1$ and $\phi_x = 0$ otherwise, *i.e.*, the state that puts all probability amplitude on 01001.

The Hadamard gate H works on one bit, and is defined as:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

$$H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

Equivalently, we can define it using matrix notation, as

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

If we have n bits in the state, then we have Hadamard gates H_1, H_2, \dots, H_n , each operating on a different bit. This is what H_1 does:

$$H_1 |0b_1b_2 \dots b_n\rangle = \frac{1}{\sqrt{2}}(|0b_1b_2 \dots b_n\rangle + |1b_1b_2 \dots b_n\rangle)$$

$$H_1 |1b_1b_2 \dots b_n\rangle = \frac{1}{\sqrt{2}}(|0b_1b_2 \dots b_n\rangle - |1b_1b_2 \dots b_n\rangle)$$

Try writing down H_1 has a 2^n -by- 2^n matrix, and you'll see why this notation is useful.

The CNOT gate is defined as:

$$\begin{aligned} \text{CNOT } |00\rangle &= |00\rangle \\ \text{CNOT } |01\rangle &= |01\rangle \\ \text{CNOT } |10\rangle &= |11\rangle \\ \text{CNOT } |11\rangle &= |10\rangle \end{aligned}$$

The CNOT gate is a classical gate, in the sense that it only changes some definite states into other definite states. In order for the operation to be unitary, all possible states have to appear on the right hand sides, *i.e.*, it wouldn't be valid to have an operation with $M|00\rangle = |00\rangle$ and $M|01\rangle = |00\rangle$, as this wouldn't be unitary.

The CNOT gate flips the second bit if the first bit is 1. Similarly, there is the CCNOT gate, which operates on 3 bits, and flips the third bit if both the first and second bits are 1. Like with to the Hadamard gate, if we have n bits we have CNOT_{ij} and CCNOT_{ijk} gates, operating on those bits. The Hadamard and CCNOT gates are a universal set of gates, which means that any unitary 2^n -by- 2^n matrix can be arbitrarily closely approximated as a product of the H_i and the CCNOT_{ijk} gates.

Thus, a we input into the quantum computer a list of operations, e.g.

$$U = H_1 \cdot \text{CNOT}_{12} \cdot H_2 \dots H_4$$

and an initial state x , and the quantum computer will sample for us a random final state y with probability $p_y = |(U|x\rangle)_y|^2$.

0.5 The Deutsch-Jozsa algorithm

TODO

References