Lecturer: Yongshan Ding *Scribe: Jonathan Baker*

Lecture 04: More Basics

October 11, 2018

1 Transformation Rule Revisited

Recall from last time, a quantum logic gate or "transformation" was defined on a single qubit to be a norm-preserving linear transformation which can be given by a unitary matrix. Specifically, when given some input quantum state, i.e. some superposition of the basis states, we want the transformation on this state out output some other valid quantum state. If we have some transformation $U : |\psi\rangle \to U|\psi\rangle$, we require $|\langle\psi|\psi\rangle|^2 = 1 = \left|\langle\psi|U^\dagger U|\psi\rangle\right|^2$. This implied $U^\dagger U = I$, which is precisely the definition of a unitary matrix. Such transformations are reversible (since unitary matrices are invertible) are deterministic, unlike measurement operators which are probabilistic and irreversible.

1.1 Single Qubit Gates

We will now explore a few important single qubit gates. The first of which is the Not gate.

Example 1.1. *The NOT Gate (X Gate). This gate is a close analog to how a classical negation operator works. We want to send* $|0\rangle \to |1\rangle$ *and* $|1\rangle \to |0\rangle$*. However, a quantum state may be in a superposition of these two states. So more generally if* $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ *then* $X|\psi\rangle = \beta|0\rangle + \alpha|1\rangle$*. Since each operation has a corresponding unitary matrix, we may verify that*

$$X \equiv \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

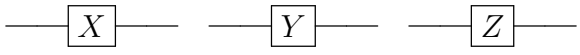
does exactly as desired by noting

$$X|\psi\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \beta \\ \alpha \end{bmatrix}$$

Example 1.2. *The X gate is just one of several so-called Pauli gates. The others are the Y and Z gates where Y and Z are defined as*

$$Y \equiv \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad Z \equiv \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

In order to construct large circuits to perform computation on qubits without having to consider the matrices explicitly, we will write circuit diagrams so that each qubit in our system is given a horizontal line parallel to each other and gates on qubits are boxes which abstractly represent an operation applied to that qubit. In these circuits, there is an implicit x-axis representing time, so that the order of gates is completely specified. For the single qubit systems we've considered so far, we can represent the three Pauli gates with the following circuit diagrams:



We will now return to other single qubit gate examples.

Example 1.3. *The Hadamard Gate (H Gate). Recall from before there are many different bases for a single qubit two of which are the* $\{\ |0\rangle,\ |1\rangle\ \}$*, or the computational basis state and another being the* $\{\ |+\rangle,\ |-\rangle\ \}$*. It is often advantageous to switch between these two bases, which is precisely what a Hadamard gate can do. That is, we send* $|0\rangle \to |+\rangle$ *and* $|1\rangle \to |-\rangle$*. From this, we can see that the following matrix does precisely this*

$$H \equiv \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Notice that if given a pure $|0\rangle$ *or* $|1\rangle$ *state, the application of a Hadamard gate will produce an equal superposition of these two states.*

Example 1.4. *The Phase Gate (S Gate). Suppose we have some state* $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ *then the application of an S gate produces* $|\psi\rangle = \alpha\,|0\rangle + i\beta\,|1\rangle$*. Notice that the application of a measurement operator to the before and after state gives values* $|0\rangle$ *and* $|1\rangle$ *with the same probability. A matrix performing this operation is given by*

$$S \equiv \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$$

All of the single qubit gates mentioned up to now belong to the Clifford Gates, the set of efficiently (in polynomial time) classically simulatable gates. However, the Clifford Group is not universal for quantum computation. Universality can be achieved by adding the $T = \sqrt{S}$ gate (we often see this written as Clifford+T), which is simply a $\pi/4$ rotation about the Z axis in the Bloch Sphere. We will end with a final example of single qubit gates, arbitrary rotations.

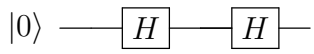
Example 1.5. *Rotation Gates* $(R_z, R_x, R_y$ *Gates). If we imagine our quantum state as a vector in the Bloch sphere, we may perform rotations (of various angles) of this vector about various axes in the sphere of, for example the X or Z axis. We may write* $R_x(\theta)$ *to indicate a rotation of* θ *degrees about the X-axis. Several of the gates we've already discussed are just examples of the* $R_z(\theta)$ *gates, specifically the Z, S, and T gates which rotate 180, 90, and 45 degrees respectively.*

It can be shown, though not important here, that, for example, $R_x(\theta) = \cos(\frac{\theta}{2})I - i\sin(\frac{\theta}{2})X$*, where* I *is the identity matrix and* X *is the NOT gate from before. Together this gives the following matrix*

$$R_x(\theta) \equiv \begin{bmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}$$

1.2 Feynman Path Sum

Consider first the following circuit, for which it isn't hard to see that the expected outcome is the original $|0\rangle$ back.



Suppose we wanted to figure out the outcome for any quantum circuit, to do this, we may used what is called the Feynman Path Sum. To see this, we will illustrate with solving the example circuit above.

Example 1.6. *Because we begin in* $|0\rangle$*, we have* $\alpha = 1$ *and* $\beta = 0$*, so the only paths which contribute to the sum come from the branches beginning in the state* $|0\rangle$*. Each layer of the tree corresponds to a collection of parallel operations in our circuit. Here, we have a single qubit and two operations with a starting state. Therefore, we have three layers. Each arrow in the tree is labeled with the effect on the amplitudes of the state based on the operation. To find the total state, we first compute the product of the values along every path from root (left) to leaf (right). Here we have four paths which give products* $\frac{1}{2}|0\rangle, \frac{1}{2}|1\rangle, \frac{1}{2}|0\rangle, \frac{-1}{2}|1\rangle$*. Summing over all paths, we get in net* $|0\rangle$*. So we see that this circuit is actually the identity.*

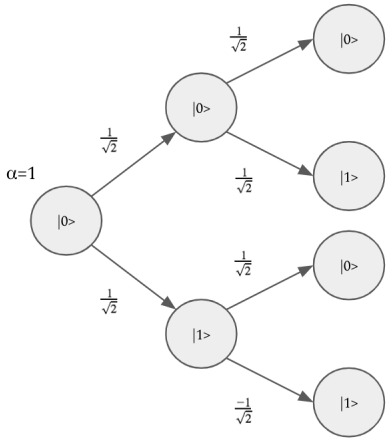


Figure 1: Feynman Path Sum branches beginning with $\alpha = 1$.

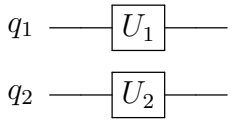
In general, the Feynman path sum is quite useful for things like simulation, optimization, and verification in quantum computing. A nice benefit of this method, which takes exponential time to compute (there are an exponential number of paths), is that it can be parallelized easily on classical processors. Furthermore, in computing the path, because it is a product, if a branch becomes relatively negligible while computing the path its outcome can be discarded and a new path can then be computed. This prevents us from having to compute every path in the tree at the cost of some error in the final result.

This method also demonstrates an important quantum computing concept, namely interference. From the prior example, we saw that the two paths contributing to $|0\rangle$ were both positive amplitudes, resulting in positive/constructive interference while the two paths resulting in a $|1\rangle$ had opposing signs, resulting in destructive interference.

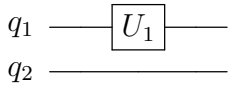
2 Multiple Qubits

Recall the joint state of two qubits $|\psi_1\rangle$ and $|\psi_2\rangle$ can be written as $|\psi_1\rangle \otimes |\psi_2\rangle$, i.e. the tensor product of the two states. One can quickly check that the dimension of this joint state has dimension 4 resulting in an exponential growth in the number of states as the number of qubits in the system increases.

It can be shown, but not proven here, that the tensor product of two unitary matrices is still unitary. This means that some operations on states containing more than one qubit can be written as the tensor as operations on the individual qubits. For example, consider U_1 and U_2 single qubit operations on q_1 and q_2, then $U_1 \otimes U_2$ operates on $q_1 \otimes q_2$. The circuit for $U_1 \otimes U_2$ is written as



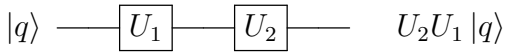
If instead our circuit looks like



we may write this operation as $U_1 \otimes I$, where I is the identity matrix, or the "no-op".

2.1 Sequantial and Parallel Execution of Quantum Gates

Consider a quantum circuit in which two operations are applied serially to a qubit q such as

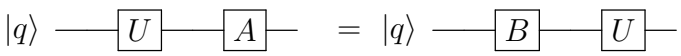


Here we apply the operation U_1 first and then apply the operation U_2. This results in a quantum state $U_2U_1 |q\rangle$, where the order is reversed because of the order of matrix multiplication.

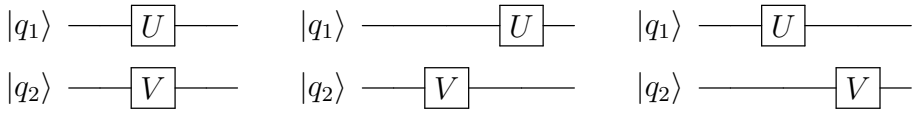
There are two criteria which can be used to determine whether operations may be reordered. The first is the *Commutation Relation*. This is defined for two gates A, B as

$$[A, B] = AB - BA \tag{1}$$

When the two gates commute then the commutator $[A, B] = 0$ which means we are free to reorder the gates, i.e. $AB |\psi\rangle = BA |\psi\rangle$. The second is the *Conjugation Relation*. Two gate A, B are conjugate if there exists another gate U so that $UAU^\dagger = B$ or equivalently $UA = BU$. As a circuit we see that we can move A forward in the circuit across some other element U by conjugating it to B first i.e.



On the other hand, we can consider parallel execution of operations on qubits. Consider the following circuits; they are all the same and either operations can be performed first



and we can check that $U \otimes V = (I \otimes V)(U \otimes I) = (U \otimes I)(I \otimes V)$. Therefore, when performing temporal scheduling of gates on qubits, we can choose to perform them in any order with no effect on the outcome. If the hardware supports parallel execution of gates, they could be performed at the same time.

2.2 Multi-qubit Gates

There are some gates which cannot be expressed as the tensor product of operations on single qubits. This is analagous to the fact that in classical computation two operand gates cannot be achieved purely with single operand gates. The first of which we will explore is the CNOT or Controlled-Not gate.

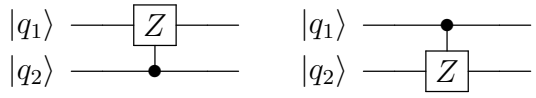
Example 2.1. *The CNOT, or Controlled-Not, operation is a two input two output gate which transforms input basis states as*

$$00 \to 00$$
$$01 \to 01$$
$$10 \to 11$$
$$11 \to 10$$

We can explain this as CNOT performs a NOT operation on the second bit only when the control (first) bit is 1. As a matrix we have

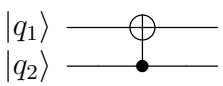
$$CNOT \equiv \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

In general, we may control any single qubit gate U. Notice that $C-U$ is **not** the same as $I \otimes U$, i.e the class of controlled gates cannot be written as the tensor of two single qubit operations as before. Another common gate is the CZ gate which behaves similarly to the CNOT gate, but instead of a NOT operation we perform a Z rotation when the control is 1. The CZ has a special property such that the following two circuits are equivalent

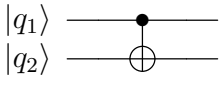


while in general this is not the case. Consider switching the control and target of the CNOT gate. Comparing the matrices for these two operations shows they are not the same

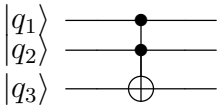
$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \equiv$$



$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \equiv$$



These gates may be controlled on more than a single control. One of the most famous examples is the Toffoli Gate (CCNOT or Controlled-Controlled-Not). It has the following circuit



The Toffoli gate can be used to achieve irreversible classical operations like AND and OR in quantum computing. One final gate, important in architectures which require qubits to be adjacent in order to perform multiqubit operations, is the SWAP gate, which switches the states of two qubits.