**Recap & Bridge: From Simple Vectors to Dirac Notation**

Recall that a single qubit state was introduced using simple vector notation, much like what you might have seen in high school linear algebra.

In quantum computing the state of a single qubit, often denoted as , was represented as a two-component column vector(computational basis state):

Here, and are complex numbers representing the probability amplitudes of finding the qubit in the:

basis state corresponding to

or the basis state corresponding to

The fundamental probability rule ensures this is a valid quantum state and requiring that the sum of the absolute squared magnitudes of the amplitudes must equal one:

Recall note:  for  is complex conjugate of

Now here comes the question. What about 2 qubits? 3 qubits, 4 qubits and so on… Are they the same representation as single qubit system in a column vector? How does the gate operation act on the multiple qubit systems?

First, let us take a look at two qubit system:

A two qubit system can be represent as:

Where  are complex numbers representing the probability amplitudes of finding the two qubit system in different state. Here, you might notice that the number of different states is 4 and this can be explain in simple. When we combine two qubit system with each qubit system has 2 states  then the different combinations we get is  and the most conventional ways of each of the complex numbers representing the corresponding probability amplitudes of each of the states is:

basis state corresponding to

basis state corresponding to

basis state corresponding to

or the basis state corresponding to

So, in the context sense of probability, it is also similar with the single qubit system where the absolute squared of these four probability amplitudes must equal to 1:

By showing the example of 2 qubit system by referencing single qubit system intuitively, we can also easily guess the other higher number qubit system. But first, we need a more rigorous mathematical language to express the entire qubit system in general form.

**Dirac notation**

Dirac notation, also known as a Bra-Ket notation, is a mathematical framework developed by physicist Paul Dirac. It is the standard, concise and abstract language used in quantum mechanics and quantum computing to represent states, vectors and operations in a complex vector space (Hilbert space). It represents the abstract quantum state of a system (e.g., a qubit)

The notation gets its name from the fact that an inner product (or dot product) of two vectors is formed by combining a "bra" and a "ket" to form a "bracket":

**\*\*Note** that Bra-Ket notation acts like a variable you use in high school, so

As you might notice, the notation that we use from the start are Dirac notation and we are using this notation to help us write out the calculation and equations in Quantum Computing in a simpler and tidy way.

**Checkpoint**

What is the possible combinations of a 4 qubit system with as computational basis state?

**Additional topic**

**Tensors product in multiple qubit system**

We saw that the column vector of 2 qubit system has 4 entries in it. The correct way to formulate this vector(rather than intuitively counting like the section before) is through tensor product. For instance:

Thus, in a 2qubit system we have a 4 dimension column vector. This tensor structure explains why the number of basis states doubles with each added qubit(qubit system dimensions grow by where is the number of qubit in the system).

**Checkpoint**

Compute and write it as a column vector. What is the dimension of the state space for 3 qubits?

**The Ket Vector:**

To write out the qubit system with  different states, it is commonly written using the column matrix notation(ket notation):

or

where for each  we defined as the basis state of the qubit system. For example in the 2 qubit system:

This is one of the important characteristics of using vectors, where they can be defined and express as a set of vectors that are both linearly independent and can span the entire Hilbert space.

\*\*This expression is similar to the Cartesian system we use in representing coordinates, but do not confuse with the coordinate system used with the qubit system as the vector space expression use in quantum computing is to represent the quantum state of the qubit.

**The Bra Vector:**

The bra notation, represents the conjugate transpose (often called the Hermitian conjugate or ) of the corresponding ket. It is written in row matrix notation.

or

**Checkpoint:**

For a vector determine

**Inner product:**

As you might know, inner product of the bra-ket notation, has the same operation as the inner product you may have learned in your high school linear algebra classes. The inner product is formed by combining a bra with a ket Mathematically, it is the generalization of the vector dot product for Hilbert spaces.

The result of the inner product is always a single scalar (complex or real number).

For example, Given two state vectors, and

The inner product is calculated by multiplying corresponding elements and summing the results:

If you made it to this part, you are almost completely understand the basic building blocks of learning Quantum Computing, the next section will try to explains the significant meaning of the bra-ket notation and its probability rule.

**Checkpoint:**

For a vector determine . Is the answer equal to 1(normalized)?

Then, determine  what does the answer mean?

**Born’s rule**

In quantum mechanics, a system's state is described by a complex vector, which contains probability amplitudes. But, these amplitudes are *not* probabilities themselves. So a physicist, Max Born states that the probability is equal to the absolute square of the magnitude of the complex number coefficient (or "amplitude") associated with that state in the wave function. Hence, the probability rule we keep obey in the previous notes. Let us look back at the example before:

and

Born’s rule gave a proper meaning of the amplitudes of different states of the a quantum system. Before Max Born devised this rule, Erwin Schrödinger developed the famous Schrödinger Equation, which uses a mathematical function to describe the quantum state over time. The problem was, no one knew what itself *was* physically. This simple rule provided the necessary bridge between the abstract, wavy math and the discrete, particulate results seen in the lab. It explained why experiments yielded statistical averages that matched the wave function's shape, even though individual measurements always returned a single, definite outcome.

Thus, for a valid quantum systems to represent a qubit, one need to make sure the qubit system is normalized, and this where inner products come in handy. For example:

This means that  is a normalized state to describe a qubit system in the Hilbert space. But we live in a chaotic world, so we have to use Born’s rule to help us in giving the meaning what we measure from nature. In general, an unnormalized state:

To normalized:

By doing this, we ensure that the summation of square of absolute of the probability amplitudes equals , which yield the probabilistic meaning and statistical interpretation of the results.

**Checkpoint:**

Suppose you have an unnormalized two-qubit state

Find the normalized  state.

Another thing that we can extract info from using inner product in Born’s rule framework is finding the probability of different state in the qubit. For example, a qubit state:

Recall that the qubit state above can express in terms of 2 qubit computational basis state :

We can see here that the probability amplitude of  is  and the probability of getting this state is . With inner product, we can extract the probability amplitude of the qubit state:

By performing Inner product with any basis state that exist in the Hilbert space which is defined carefully, we can easily get the probability amplitude of that basis state.

This simple case shows the significance of using inner product that helps to build an intuitive sense, this is important when we get more and more complex as we continue to progress in further chapter.

**Outer product**

We have thus formulated the basic language of quantum computing using state vectors, there is also another alternate ways to formulate known as density operator. Let us take a look at the equation below:

The **Outer Product**, written as is the reverse multiplication of the inner product. Instead of resulting in a single scalar, the outer product of a ket (column vector) and a bra (row vector) results in a **square matrix (an operator)**.

For any two state vectors, the generally outer product is calculated as matrix multiplication:

The next section that we will focus on is when the two states are the same, the resulting matrix acts as a **Projection Operator**, which projects any state onto the subspace. Most importantly, this specific outer product forms the **Density Matrix** for a **Pure State**.

Before we discuss on **Density Matrix**  **Projection Operato**r is a fundamental concept in quantum mechanics that bridges the mathematical state of a system (the vector) with the physical action of observation (the measurement).

**The Projection Operator**

Mathematical Definition

A **Projection Operator** (usually denoted or ) is a special type of matrix (operator) that satisfies the property of **idempotence:** . Applying the operator once is the same as applying it multiple times. Why? Because once a vector is projected onto a subspace, projecting it again doesn't change its position.

In the context of Dirac notation, the projection operator for a normalized state is defined as the outer product:

Modelling Measurement In quantum mechanics, projection operators are essential because they mathematically model the process of **measurement.** When you measure a quantum state with respect to a basis state you are performing a projection. The operator extracts the component of that lies along the direction of When you apply the projection operator to a general state

Since is the inner product (a scalar number, the probability amplitude), the result simplifies to:

The result of the projection is a new vector that is entirely in the ∣k⟩ direction, scaled by the amplitude of the original state.

**Simple Example**: Projection onto . For a single qubit, the project operator onto the state is

Applying to a general state

The operator successfully **projects** the vector by keeping the amplitude (the component) and eliminating the component (the component).

The purpose of outer product and the projection operator serves as an alternative ways to extract the states that we need from the qubit consists of superposition of different states.

Now, we are ready to move forward to the density matrix.

**Noise Integration: Formal Models (Pure vs. Mixed States)**

In the real world, quantum computers are highly sensitive to the environment, leading to **decoherence**. This noise forces us to move beyond the simple state vector description.

**The Ideal State: Pure State**

The state vector describes a **Pure State**, where we have maximum knowledge about the system (i.e., we know its exact superposition). It assumes the qubit is perfectly isolated and noiseless.

**The Noisy State: Mixed State (Density Matrix )**

A **Mixed State** describes a quantum system that is a statistical ensemble (probabilistic mixture) of multiple pure states due to noise. We use the **Density Matrix** ( pronounced "rho") to describe this state, which naturally accounts for uncertainty and environmental decoherence.

**Ideal Pure State Density Matrix:** For a pure state the density matrix is simply the outer product:

**Noisy Mixed State Density Matrix:**

For a mixed state (a probabilistic blend of states with probabilities ):

The density matrix provides two distinct pieces of information about the quantum state: 1.

**Diagonal Elements** These elements always give the measurement probabilities: They represent the statistical likelihood of observing the system in basis state

**Off-Diagonal Elements** These elements, often called **coherence terms** or **cross terms**, track the phase relationship between basis states and Their non-zero value is the mathematical signature of **superposition** and **entanglement**. Noise causes these terms to decay to zero, a process known as **decoherence**, turning the quantum state into a classical mixture.$$