Quantum Circuits

Deval Deliwala

December 24, 2023

Contents

This lesson introduces the *quantum circuit* model of computation, a standard description of quantum computations we'll use further onwards.

We will also introduce a few important mathematical concepts including *inner products*, the notion of *orthogonality* and *orthonormality*, and *projections* and *projective measurements*, which generalize standard basis measurements. Through these concepts, we'll derive fundamental limitations on quantum information, including the *no-cloning theorem* and the impossibility to perfectly discriminate non-orthogonal quantum states.

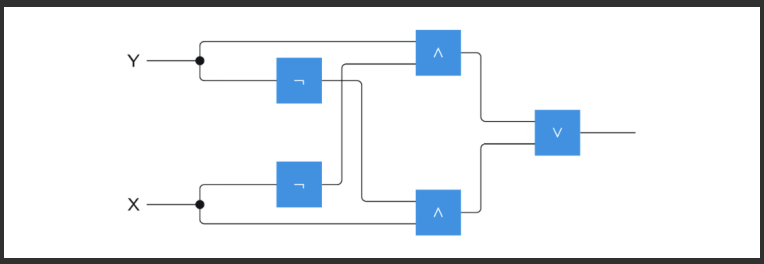
1 Circuits

In computer science, *circuits* are models of computation in which information is carried by wires via a network of *gates*, which represent operations that transform the information carried by the wires.

Although the word "circuit" often refers to a circular path, circular paths aren't actually allowed in most circuit models of computation. Quantum circuits follow this pattern – a quantum circuit represents a finite sequence of operations that cannot contain feedback loops.

1.1 Boolean Circuits

Here is an example of a classical Boolean circuit, where wires carry binary values and the gates represent Boolean logic operations:



XOR

The flow of information along the wires goes from left to right, the wires on the left-hand side of the figure labeled X and Y are input bits, which can each be set to whatever binary value we choose, and the wire on the right-hand set is the output. The immediate wires take whatever values are determined by the gates, which are evaluated from left to right.

The gates are AND gates ($\wedge$), OR gates ($\vee$), and NOT gates ($\neg$).

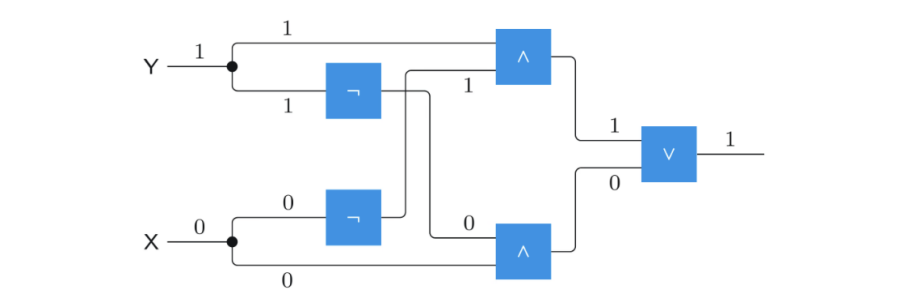
The two small circles on the wires just to the right of the names X and Y represent *fanout* operations, which simply create a copy of whatever value is carried on the wire on which they appear,

so that this value can be input into multiple gates.

The particular circuit in this example computes the XOR, which is denoted by the symbol $\oplus$

ab	$a \oplus b$
00	0
01	1
10	1
11	0

In the next diagram we consider just one choice for the inputs: $X = 0$ and $Y = 1$. Each wire is labeled by value it carries so you can follow the operations. The output value is 1 in this case, which is the correct value for $XOR : 0 \oplus 1 = 1$.



You can check the other three possible input settings in a similar way.

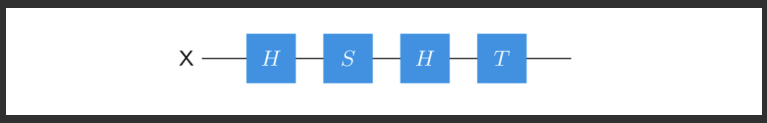
1.2 Other types of circuits

In *arithmetic circuits* the wires may carry integer values and the gates may represent arithmetic operations, such as addition and multiplication. We can also consider circuits that incorporate randomness, such as ones where gates represent probabilistic operations.

1.3 Quantum Circuits

In the quantum circuit model, wires represent qubits and gates represent operations acting on these qubits. We'll focus for now on operations we've encountered so far, namely *unitary* operations and *standard basis measurements*.

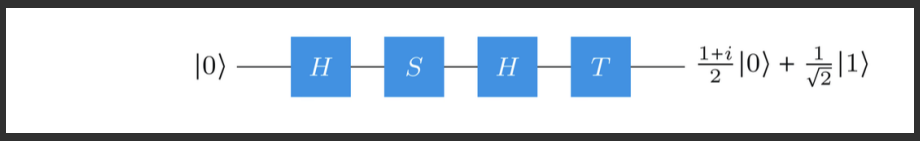
Here is a simple example of a quantum circuit:



In this circuit we have a single qubit X and a sequence representing unitary operations on this qubit. Just like in the example above, the flow of information goes from left to right. The first

operation is Hadamard, then S, then another Hadamard, and a final T operation. Applying the entire circuit therefore applies the composition of these operations, $THSH$, to the qubit X.

Sometimes we wish to explicitly indicate the input or output states to a circuit. For example, if we apply the operation $TSHS$ to the state $|0\rangle$, we obtain the state $\frac{1+i}{2}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$. We may indicate this as follows:



Quantum circuits often have all qubits initialized to $|0\rangle$, but there are also cases where we wish to set the input qubits to different states.

Here is how we can specify this circuit in Qiskit:

[4]:

```
from qiskit import QuantumCircuit

circuit = QuantumCircuit(1)

circuit.h(0)
circuit.s(0)
circuit.h(0)
circuit.t(0)

circuit.draw()
```

[4]:

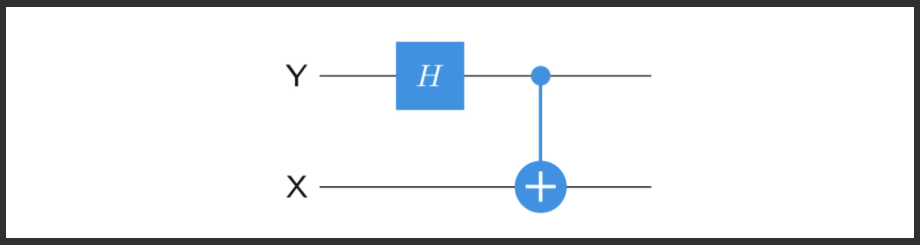
The default names for qubits in Qiskit is q_0, q_1, q_2, etc., and when there is a single qubit it is q. If we wish to choose our own name we can do this using the `Quantum Register` class like this:

[6]:

```
from qiskit import QuantumRegister

X = QuantumRegister(1, "X")
circuit = QuantumCircuit(X)

circuit.h(X)
circuit.s(X)
circuit.h(X)
circuit.t(X)

circuit.draw()
```

[6]:

Here is another example of a quantum circuit, this time with two qubits:



The second gate is a two-qubit gate – a CNOT operation, where the solid circle represents the control qubit and the circle with a plus inside denotes the target qubit.

Before examining this circuit in greater detail and explaining what it does, it is important to clarify how qubits are ordered in quantum circuits

Ordering of Qubits in Quantum Circuits

The topmost qubit in a circuit has index 0, and corresponds to the rightmost position in a Cartesian or Tensor Product. The second-to-top has index 1 and corresponds to the position second-from-right in a Cartesian or Tensor Product, and so on down to the bottom-most qubit which has the

highest index, and corresponds to the leftmost position in a Cartesian or Tensor product.

So for example, when we refer to a qubit in the zeroth position, we're referring to the topmost qubit in a circuit diagram or the rightmost qubit in the expressing of a quantum state vector; the qubit in the first position is second-from-top in a circuit diagram or second from right in a quantum state vector; and so on. This indexing convention is known as *little-endian* indexing, because the indexes start at the "little end" when we think about the significance of bits in binary representation of numbers.

Thus, in the circuit above, we are considering the circuit to be an operation on two qubits (X, Y). If the input is $|\psi\rangle|\phi\rangle$, then the lower qubit (X) starts in the state $|\psi\rangle$ and the upper qubit (Y) starts in the state $|\phi\rangle$.

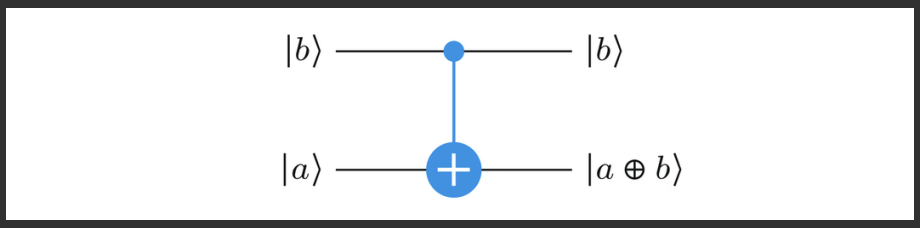
Now let's look at the circuit itself, moving from left to right through its operations.

1. The first operation is a Hadamard Operation on Y.

When applying a gate to a single qubit like this, nothing happens to the other qubits; nothing happening is equivalent to the identity operation. In our circuit there is just one other qubit X, so the Hadamard Operation represents this operation:

$$I \otimes H = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}$$

2. The second operation is the controlled-NOT (CNOT) operation, where Y is the control and X is the target. The CNOT gate's action on standard basis sates is as follows:



Given that we order the qubits as (X, Y), the matrix representation of the CNOT gate is this:

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

The unitary operation of the entire circuit, which we'll call U is the composition of the two operations:

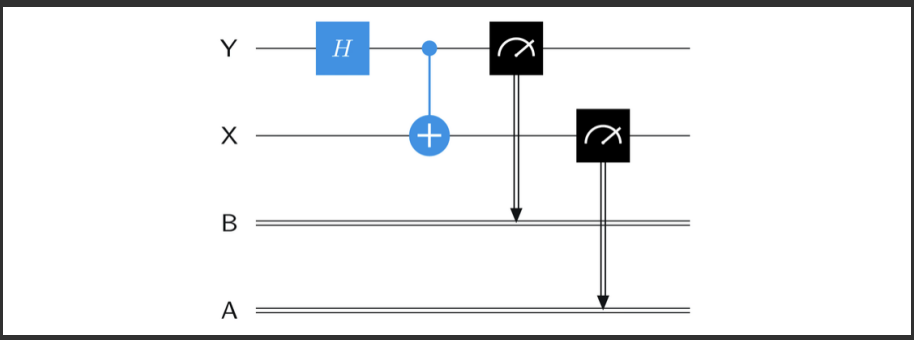
$$U = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 \end{pmatrix}$$

Recalling our knowledge of Bell states, we get that

$$\begin{aligned} U|00\rangle &= |\phi^+\rangle \\ U|01\rangle &= |\phi^-\rangle \\ U|10\rangle &= |\psi^+\rangle \\ U|11\rangle &= -|\psi^-\rangle \end{aligned}$$

This circuit gives us a way to convert the standard basis into the bell basis. (The -1 phase factor on the last state could be eliminated if we wanted by adding a controlled-Z gate at the beginning or a swap gate at the end, for instance).

In general quantum circuits can contain any number of qubit wires. We may also include classical bit wires which are indicated by double lines:

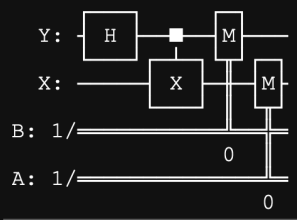


In this circuit we have a Hadamard Gate and a controlled-NOT gate on two qubits X and Y, just like in the previous example. We also have two *classical* bits, A and B, as well as two measurement gates. The measurement gates represent standard basis measurements: the qubits are changed into their post measurement states, while the measurement outcomes are overwritten onto the classical bits to which the arrows point.

Here ins an implementation of this circuit using Qiskit:

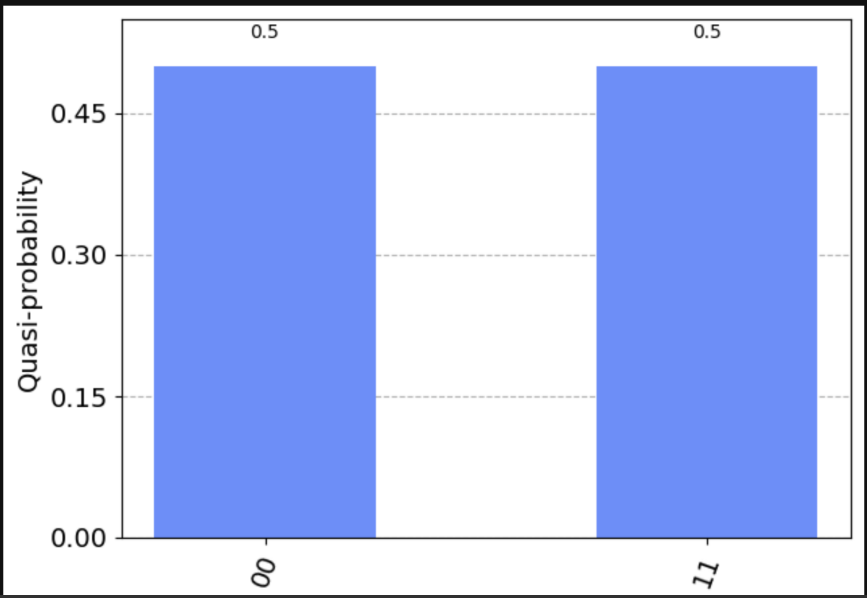
[7]:
```
from qiskit import QuantumCircuit, QuantumRegister, ClassicalRegister

X = QuantumRegister(1, "X")
Y = QuantumRegister(1, "Y")
A = ClassicalRegister(1, "A")
B = ClassicalRegister(1, "B")

circuit = QuantumCircuit(Y, X, B, A)

circuit.h(Y)
circuit.cx(Y, X)
circuit.measure(Y, B)
circuit.measure(X, A)

circuit.draw()
```

[7]:



[8]:
```
from qiskit.primitives import Sampler
from qiskit.visualization import plot_histogram

sampler = Sampler()
results = sampler.run(circuit).result()
statistics = results.quasi_dists[0].binary_probabilities()

plot_histogram(statistics)
```

[8]:



Sometimes it is convenient to depict a measurement as a gate that takes a qubit as input and outputs a classical bit (as opposed to outputting the qubit its post-measurement state and writing the result to a separate classical bit. This means that the measured qubit has been discarded and can be safely ignored thereafter.

For example, the following circuit diagram represents the same process as the one in the previous diagram, but where we ignore X and Y after measuring them.

The default names for qubits in Qiskit is q_0, q_1, q_2, etc., and when there is a single qubit it is q. If we wish to choose our own name we can do this using the `Quantum Register` class like this:

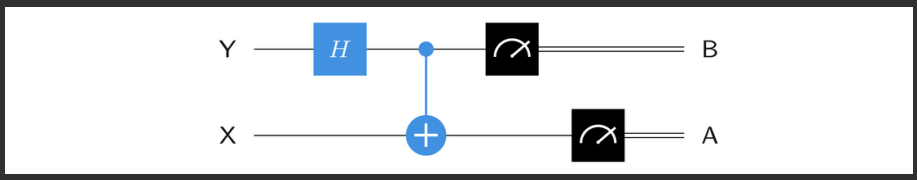
[6]:

```
from qiskit import QuantumRegister

X = QuantumRegister(1, "X")
circuit = QuantumCircuit(X)

circuit.h(X)
circuit.s(X)
circuit.h(X)
circuit.t(X)

circuit.draw()
```

[6]:

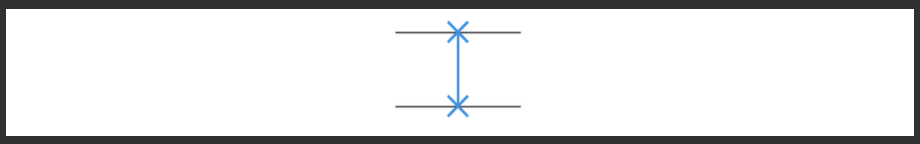
1.4 Diagrams of Qubit Gates

Single-Qubit gates are generally shown as squares with a letter indicating which operation it is, like this:

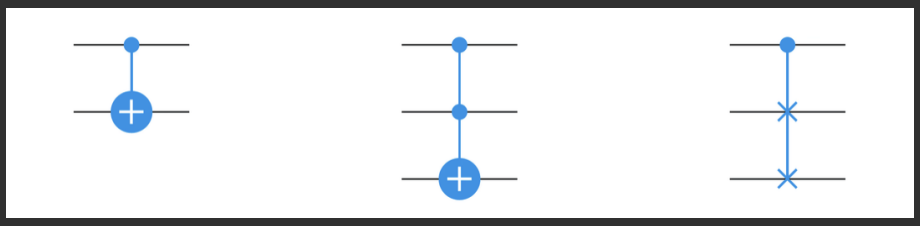


NOT (or X) gates are also sometimes denoted by a circle around a + sign.
Swap gates are denoted as follows:

Controlled-gates, meaning that gates that describe controlled-unitary operations, are denoted by a filled-in circle (indicating the control) connected by a vertical line to whatever operation is being controlled. For instance, controlled-NOT gates, controlled-controlled-NOT gates, and controlled-swap gates are denoted like this:



Arbitrary unitary operations on multiple qubits may be viewed as gates. They are depicted by rectangles labeled by the name of the unitary operation. For instance, here is a depiction of an (unspecified) unitary operation U as a gate, along with a controlled version of this gate:



1.5 Inner Products, Orthonormality, and Projections

The notions of the *inner product*, *orthogonality*, and *orthonormality* for sets of vectors, and *projection* matrices will allow us to introduce a handy generalization of standard basis measurements.

1.6 Inner Products

We refer to an arbitrary column vector as a ket, such as

$$|\psi\rangle = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{pmatrix}$$

the corresponding bra vector is the *conjugate transpose* of this vector:

$$\langle\psi| = (|\psi\rangle)^\dagger = \begin{pmatrix} \bar{\alpha}_1 & \bar{\alpha}_2 & \cdots & \bar{\alpha}_n \end{pmatrix}$$

Alternatively if we have some classical state set Σ in mind, and we express a column vector as a ket, such as

$$|\psi\rangle = \sum_{a\in\Sigma} \alpha_a |a\rangle$$

then the corresponding bra vector is the conjugate transpose

$$\langle\psi| = \sum_{a\in\Sigma} \bar{\alpha}_a \langle a|$$

We also observed that the product of a bra and ket vector, viewed as matrices either having a single row or a single column, results in a scalar. Specifically, if we have two (column) vectors

$$|\psi\rangle = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{pmatrix} \quad \text{and} \quad |\phi\rangle = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix}$$

then,

$$\langle\psi|\phi\rangle = \langle\psi||\phi\rangle = \begin{pmatrix} \bar{\alpha}_1 & \bar{\alpha}_2 & \cdots & \bar{\alpha}_n \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} = \bar{\alpha}_1\beta_1 + \cdots + \bar{\alpha}_n\beta_n$$

Alternatively, if we have two column vectors that we have written as

$$|\psi\rangle = \sum_{a\in\Sigma} \alpha_a |a\rangle \quad \text{and} \quad |\phi\rangle = \sum_{b\in\Sigma} \beta_b |b\rangle$$

then,

Inner Product Definition

$$\begin{aligned} \langle\psi|\phi\rangle &= \langle\psi||\psi\rangle \\ &= \left(\sum_{a\in\Sigma} \bar{\alpha}_a \langle a|\right)\left(\sum_{b\in\Sigma} \beta_b |b\rangle\right) \\ &= \sum_{a\in\Sigma}\sum_{b\in\Sigma} \bar{\alpha}_a\beta_b \langle a|b\rangle \\ &= \sum_{a\in\Sigma} \bar{\alpha}_a\beta_a \end{aligned}$$

where the last equality stems from the observation that $\langle a|a\rangle = 1$ and $\langle a|b\rangle = 0$ for classical states a and b satisfying $a \neq b$.

The value $\langle\psi|\phi\rangle$ is called the *inner product* between vectors $|\psi\rangle$ and $|\phi\rangle$. Inner products are critically important in quantum information and computation.

Let us go over some basic facts about the inner product.

Relationship to the Euclidean Norm

The inner product of any vector with itself,

$$\langle\psi|\psi\rangle = |||\psi\rangle||^2.$$

Thus, the euclidean norm of a vector may be expressed as

$$|||\psi\rangle|| = \sqrt{\langle\psi|\psi\rangle}$$

And since the euclidean norm can only equal 0 if every entry is equal to 0, for every vector $|psi\rangle$ we have

$$\langle\psi|\psi\rangle \geq 0$$

Conjugate Symmetry

For any two vectors

$$|\psi\rangle = \sum_{a\in\Sigma} \alpha_a|a\rangle \quad \text{and} \quad |\phi\rangle = \sum_{b\in\Sigma} \beta_b|b\rangle$$

we have

$$\langle\psi|\phi\rangle = \sum_{a\in\Sigma} \bar{\alpha}_a\beta_a \quad \text{and} \quad \langle\phi|\psi\rangle = \sum_{a\in\Sigma} \bar{\beta}_a\alpha_a$$

and therefore

$$\overline{\langle\psi|\phi\rangle} = \langle\phi|\psi\rangle$$

Linearity in the Second Argument (and conjugate linearity in the first)

Let us suppose that $|\psi\rangle, |\phi_1\rangle, and |\phi_2\rangle$ are vectors and α_1 and α_2, are complex numbers. If we define a new vector

$$|\phi\rangle = \alpha_1|\phi_1\rangle + \alpha_2|\phi_2\rangle,$$

then

$$\langle\psi|\phi\rangle = \langle\psi|(\alpha_1|\phi_1\rangle + \alpha_2|\phi_2\rangle) = \alpha_1\langle\psi|\phi_1\rangle + \alpha_2\langle\psi|\phi_2\rangle$$

The inner product is *linear* in the second argument. This can be verified either through the formulas above or simply by noting that matrix multiplication is *linear* in each argument.

Combining this fact with conjugate symmetry reveals that the inner product is *conjugate linear* in the first argument. That is, if $|\psi_1\rangle, |\psi_2\rangle,$ and $|\phi\rangle$ are vectors and α_1 and α_2 are complex numbers, and we define

$$|\psi\rangle = \alpha_1|\psi_1\rangle + \alpha_2|\psi_2\rangle$$

then

$$\langle\psi|\phi\rangle = (\bar{\alpha}_1\langle\psi_1| + \bar{\alpha}_2\langle\psi_2|)|\phi\rangle = \bar{\alpha}_1\langle\psi_1|\phi\rangle + \bar{\alpha}_2\langle\psi_2|\phi\rangle$$