Yahya Alhinai

Mar 3, 2020

EE 5340

Prepare the initial state $\boldsymbol{\Psi_0}$:

Creating the first qubit state that will be send:

$$|\Psi >= \sqrt{0.6}\ |0> + \sqrt{0.4}\,|1>$$

```
q1 = ket(sqrt(0.6),sqrt(0.4))
print(dirac(q1))

//output
[1] "0.775|0> + 0.632|1>"
```

Generating EPR pair:

$$|B_{00}>= \frac{1}{\sqrt{2}}\ |00> + \frac{1}{\sqrt{2}}\,|11>$$

```
q23 = intket(x=c(0,1,2,3),n=2,amplitudes=c(1,0,0,1))
print(dirac(q23))

//output
[1] "0.707|00> + 0.707|11>"
```

Combine them to create 3-qubit states:

$$\Psi_0 = |\Psi, B_{00}>= \sqrt{0.3}\ |000> + \sqrt{0.3}\ |011> + \sqrt{0.2}\ |100> + \sqrt{0.2}\ |111>$$

```
qu = tensor(q1,q23)
print(dirac(qu))

//output
[1] "0.548|000> + 0.548|011> + 0.447|100> + 0.447|111>"

```

Step 1 - CNOT gate:

Perform CNOT-gate on qubit 2 controlled by qubit 1. This will flip the state of qubit 2 if and only if the state of qubit one is 1.

$$\Psi_1 = \sqrt{0.3}\ |000> + \sqrt{0.3}\ |011> + \sqrt{0.2}\ |110> + \sqrt{0.2}\ |101>$$

```
qu <- tensor(CX(),I()) %*% qu
print(dirac(qu))

//output
[1] "0.548|000> + 0.548|011> + 0.447|101> + 0.447|110>"
```

Step 2 - H gate:

Perform H-gate on qubit 1. This force qubit 1 to split its probity between the state 0 and 1 that will translate into after H-gate transformation.

$$\Psi_2 = \sqrt{0.15}\ |000> + \sqrt{0.1}\ |001> + \sqrt{0.1}\ |010> + \sqrt{0.15}\ |011>$$

$$+ \sqrt{0.15}\ |100> - \sqrt{0.1}\ |101> + \sqrt{0.1}\ |110> + \sqrt{0.15}\ |111>$$

```
qu <- tensor(H(),I(),I()) %*% qu
print(dirac(qu))

//output
[1] "0.387|000> + 0.316|001> + 0.316|010> + 0.387|011> + 0.387|100> + -0.316|101> + -
0.316|110> + 0.387|111>"
```

Step 3 - Measurement:

Measure qubit 1 and qubit 2. This will make them collapse to either 0 or 1 with the following probability. Those probabilities are dependent solely on the initial state of qubit 1.

$$30\%\ \mathit{change\ will\ fall\ into\ the\ state} \rightarrow \Psi_3 = \sqrt{0.6}\ |000> + \sqrt{0.4}\ |001>$$

$$30\%\ \mathit{change\ will\ fall\ into\ the\ state} \rightarrow \Psi_3 = \sqrt{0.4}\ |010> + \sqrt{0.6}\ |011>$$

$$20\%\ \mathit{change\ will\ fall\ into\ the\ state} \rightarrow \Psi_3 = \sqrt{0.6}\ |100> - \sqrt{0.4}\ |101>$$

$$20\%\ \mathit{change\ will\ fall\ into\ the\ state} \rightarrow \Psi_3 = -\sqrt{0.4}\ |110> + \sqrt{0.6}\ |111>$$

```
L <- measure(qu, 0, 1, l2r=TRUE)
qu <- L[[1]]
print(dirac(qu))

//output is going to be one of those 4 states:
[1] "0.775|000> + 0.632|001>"
//OR
[1] "0.632|010> + 0.775|011>"
//OR
[1] "0.775|100> + -0.632|101>"
//OR
[1] "-0.632|110> + 0.775|111>"
```

Step 4 - Corrective X or Z gates:

On qubit 3, performing X-gate contorted by qubit 2 followed by Z-gate contorted by qubit 1 to produce the following:

$$\Psi_4 = \sqrt{0.6}\ |xx0 > + \sqrt{0.4}\ |xx1 >$$
$$(x\ represents\ don't\ care\ state\ here)$$

```
// X and Z gate performance depending on the previous measurement
qu <- controlled( gate=X(), n=3, cQubits=1, tQubit=2) %*% qu
qu <- controlled( gate=Z(), n=3, cQubits=0, tQubit=2) %*% qu
print(dirac(qu))

//output
[1] "0.775|000> + 0.632|001>"
//OR
[1] "0.775|010> + 0.632|011>"
//OR
[1] "0.775|100> + 0.632|101>"
//OR
[1] "0.775|110> + 0.632|111>"
```

Qubit 3 will have the initial state of qubit 1. Finalizing the teleportation of the state of qubit from one end of the quantum circuit to the other.

$$\Psi_{4,\ qubit\ 3} = |\Psi >= \sqrt{0.6}\ |0 > + \sqrt{0.4}\,|1 >$$