Quantum Random Number Generator

July 23, 2022

1 Background

Qubits have a potential to be very effective at generating random numbers. By leveraging the probabilistic nature of qubits, we could develop a kind of efficient and reliable random number generator without having to depend on the deterministic nature of pseudo-random number generating algorithms that use predictable seeds.

There are multiple approaches that we can take to implement such a program. They would need to follow the criteria:

1. Registers should be in an equal superposition of n desired states
2. The outputs of measurements should be easily convertible to numbers between $1 - n$

2 Approach: Generalized W-State

The W state is expressed as the following 3-qubit system:

$$\frac{1}{\sqrt{3}}(|001\rangle + |010\rangle + |100\rangle)$$

To construct such a state, we can make use of an arbitrary rotation gate: perhaps the R_y gate, since it doesn't have any complex phase components. For the W_3 state, we would need an equal superposition of 3 states, and hence, the normalization factor must be $\frac{1}{\sqrt{3}}$.

We would then want to construct a matrix that would get our state (partially) to where we want it to be; something along the lines of a $\frac{1}{3} : \frac{2}{3}$

probability between two states. From there, we can split our $\frac{2}{3}$ probability state into two others, perhaps by using a similar, generalized unit matrix that we used to split our state into $\frac{1}{3}$ and $\frac{2}{3}$ probability.

What would such a matrix look like? We can bear the structure of the R_y gate in mind (because we will most likely use it for our implementation of the generalization). Recall that

$$R_y = \begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}$$

We want to transform one of our qubits in our register $|\Psi\rangle = |000\rangle$:

$$|\Psi_0\rangle = |0\rangle \to \frac{1}{\sqrt{3}}(|0\rangle + \sqrt{2}\,|1\rangle)$$

Or, in matrix format:

$$U \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{3}} \begin{bmatrix} 1 \\ \sqrt{2} \end{bmatrix}$$

where U denotes some unitary gate. Luckily, the gate that would perform this operation looks something like, as derived from some basic linear equations:

$$U = \frac{1}{\sqrt{3}} \begin{bmatrix} 1 & -\sqrt{2} \\ \sqrt{2} & 1 \end{bmatrix}$$

There are several reasons why we define our gate like this. One reason is that we want our end product to look something like the R_y gate so that we can easily apply arbitrary rotations to reach our desired state. In another vein, we can reason that, since we want

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} \to \frac{\sqrt{1}}{\sqrt{3}} \begin{bmatrix} 1 \\ \sqrt{2} \end{bmatrix}$$

we would want

$$\begin{bmatrix} 0 \\ 1 \end{bmatrix} \to \frac{\sqrt{1}}{\sqrt{3}} \begin{bmatrix} -\sqrt{2} \\ 1 \end{bmatrix}$$

as well, to preserve the relative phase action of a gate on the $|1\rangle$ state. Combining these transformations on the two basis vectors generates the aforementioned linear map. Another reason why the matrix must be in the specified state is the property that all quantum gates must be unitary; a gate multiplied with its conjugate transpose must equal the identity matrix.

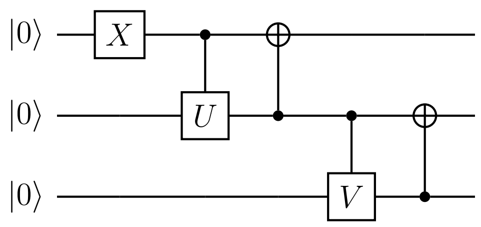
We can quickly verify that this is true.

$$\frac{1}{\sqrt{3}}\begin{bmatrix}1 & -\sqrt{2}\\ \sqrt{2} & 1\end{bmatrix}\cdot\frac{1}{\sqrt{3}}\begin{bmatrix}1 & \sqrt{2}\\ -\sqrt{2} & 1\end{bmatrix}=\frac{1}{3}\begin{bmatrix}3 & 0\\ 0 & 3\end{bmatrix}=I$$

After applying this gate, we have the state $|\Psi\rangle = \frac{1}{\sqrt{2}}(|000\rangle + \sqrt{2}\,|100\rangle)$. What we now need to do is split $\sqrt{2}\,|100\rangle$ into its own separate states; mathematically, we need to manipulate the state in such a way that $\sqrt{2}\,|100\rangle \to \sqrt{2}\left(\frac{1}{\sqrt{2}}(|110\rangle + |100\rangle)\right)$. We can do this with a controlled rotation gate on $|\Psi_0\rangle$, since the first qubit controls the further split within our manipulations. The gate we can use to do this is

$$V = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & -1\\ 1 & 1\end{bmatrix}$$

We end up with the state $\frac{1}{\sqrt{3}}(|000\rangle + |100\rangle + |110\rangle)$. While this is not exactly what we want, it is close; we have achieved the uniform superposition across 3 qubits. We shift the qubits into a state we need with X and CNOT gates; our final circuit ends up looking something like:



There seems to be some kind of pattern here. What if we generalized this to an n-qubit state? From our derivations, we already know that the normalizing coefficient should be something along the lines of $\frac{1}{\sqrt{n}}$. We can also observe that the gate used to create these specified superpositions have outputs with ratios

$$1 : n - 1$$

So, we can generalize the gate that transforms each probability ratio as:

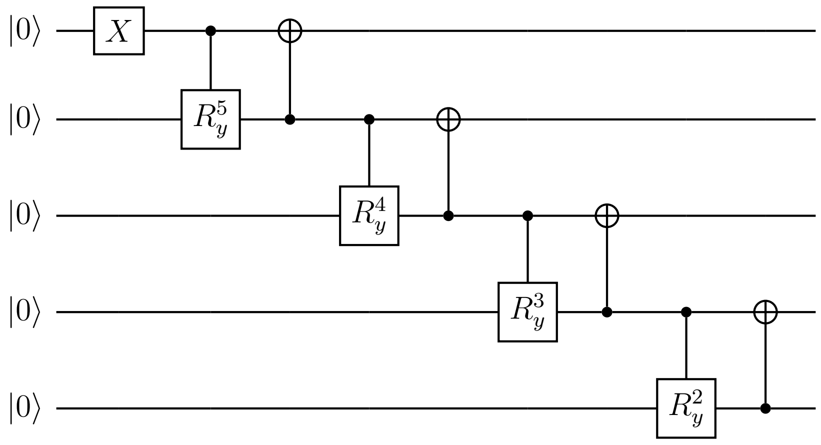
$$U(n) = \frac{1}{\sqrt{n}}\begin{bmatrix}1 & -\sqrt{n-1}\\ \sqrt{n-1} & 1\end{bmatrix}$$

We can go even further and express this in terms of the R_y gate that we have been considering. Since the cosine and sine terms match up, we would only need to find θ for one of the corresponding functions:

$$\cos\frac{\theta}{2} = \frac{1}{\sqrt{n}}$$
$$\frac{\theta}{2} = \arccos\frac{1}{\sqrt{n}}$$
$$\theta = 2\arccos\frac{1}{\sqrt{n}}$$

So, for every "split" we need to do in our superposition, we can use the R_y gate with $\theta = 2\arccos\frac{1}{\sqrt{n}}$. Furthermore, we can just apply cascading units for each qubit in our system; the usage of the CNOT gates to manipulate the bits generalizes as well. For notation purposes, we can express our "splitter" gate with an input n as R_y^n.

For example, a W_5 state could look something like:



As you can see, we just keep iterating downards from R_y^n until we reach an R_y^2 gate. After we measure the register, the position of the $|1\rangle$ in the state would thus be convertible to a useful random number. For example, $|001\rangle$ can denote 1, $|010\rangle$ can denote 2, and so on and so forth.

3 Discussion

However, the W-state approach is wasteful in using qubits; using one qubit to represent one number increases in size extremely fast and leaves many potential states unused. A better solution would be to prepare an equal superposition where only $\lceil \log_2(n) \rceil$ qubits are needed; however, I have not found a *general* circuit that completes this task.