Tomography of W-states created in log(n) steps

Johan Cattin

Project Supervisor: Dr. N. Macris

June 2019

1 Introduction

In this project, we focus on a particular type of entangled quantum states, the W states. For N qubits, $|W_N\rangle$ is the superposition of N entangled states consisting of all qubits set to $|0\rangle$ except for one qubit set to $|1\rangle$:

$$|W_N\rangle = \frac{1}{\sqrt{N}}\Big(|100...0\rangle + |010...0\rangle + |0...01\rangle\Big) = \frac{1}{\sqrt{N}}\sum_{i=1}^{N}\bigotimes_{j=1}^{N}\begin{cases}|1\rangle & \text{if } i=j,\\ |0\rangle & \text{otherwise.}\end{cases}$$

In 2018, a team of students supervised by Prof. Macris proposed an algorithm to create states $|W_N\rangle$ states in $O(log_2 N)$ time complexity [1], instead of the linear time-complexity algorithm used until then. As W states are particularly fragile in time, a faster method also leads to increased fidelity. The goal of this project is to use tomography tools provided by the open-source quantum computing framework "Qiskit" to measure the fidelity of W states created with the fast algorithm.

2 W-states generation and tomography

2.1 Generation algorithm

2.1.1 Linear time complexity

A simple approach to entangle N qubits into $|W_N\rangle$ is to start with the first qubit set to $|1\rangle$ and entangle successive pairs of qubits $(1,2),(2,3),\ldots,$ until $(N-1,N)$. For this construction, we use a sequence of controlled block $B(p)$ with $0<p<1$, $B(1/N), B(1/(N-1)), ..., B(1/3), B(1/2)$, as shown in Fig. 1.

The gate $G(p)$ corresponds to a controlled rotation around the y-axis with angle θ determined by $cos(\theta/2) = \sqrt{p}$ and $sin(\theta/2) = \sqrt{1-p}$. This rotation can be represented by the following matrix:

$$G(p) = \begin{pmatrix}\sqrt{p} & -\sqrt{1-p}\\ \sqrt{1-p} & \sqrt{p}\end{pmatrix} = \begin{pmatrix}\cos(\theta/2) & -\sin(\theta/2)\\ \sin(\theta/2) & \cos(\theta/2)\end{pmatrix}.$$

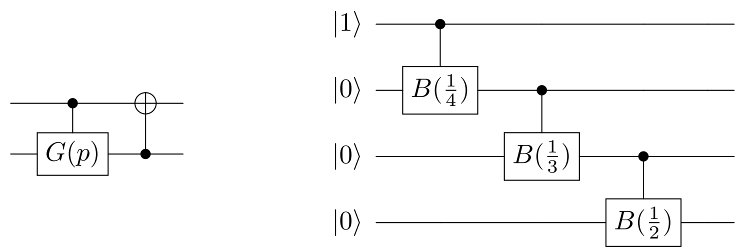


Figure 1: $B(p)$ block and circuit for $|W_4\rangle$.

We have at our disposal the IBMQ gate $R_y(\theta)$. To create a controlled $G(p)$ gate, we can perform the rotation in two steps, inserting a $CNOT$ gate after each step:

$$CG(p) = \left(I \otimes R_y\left(\frac{\theta}{2}\right)\right) CNOT \left(I \otimes R_y\left(\frac{-\theta}{2}\right)\right) CNOT.$$

The circuit is initialised in state $|1000\rangle$. After the first block, we have $\sqrt{1/4}\,|1000\rangle + \sqrt{3/4}\,|0100\rangle$. After the second,

$$\sqrt{1/4}\,|1000\rangle + \sqrt{3/4}\sqrt{1/3}\,|0100\rangle + \sqrt{3/4}\sqrt{2/3}\,|0010\rangle,$$

and the final state is:

$$\sqrt{\frac{1}{4}}\,|1000\rangle + \sqrt{\frac{3}{4}\frac{1}{3}}\,|0100\rangle + \sqrt{\frac{3}{4}\frac{2}{3}\frac{1}{2}}\,|0010\rangle + \sqrt{\frac{3}{4}\frac{2}{3}\frac{1}{2}}\,|0001\rangle$$

$$= \sqrt{\frac{1}{4}}\Big(|1000\rangle + |0100\rangle + |0010\rangle + |0001\rangle\Big) = |W_4\rangle.$$

2.1.2 Logarithmic time complexity

Figure 1 shows that the first qubit is unused in the second and third time slices. We could use the first qubit to propagate the entanglement to the fourth one in the second time slot, as those two qubits are not involved in any operation at that time. Figure 2 illustrates how to create $|W_4\rangle$ in linear time, with time slices identified by vertical dotted lines, and how to create $|W_4\rangle$ in logarithmic time, with the fourth qubit moved to first position in the circuit. When using every qubit already entangled to propagate the state, each steps doubles the number of entangled qubits, leading to a $|W_{2^m}\rangle$ state in m steps. The parameter p also needs to be adjusted: Instead of $p_i = 1/(N-i)$ used in linear-time circuits, we have $p_i = 1/2$ in $O(log_2(N))$-time circuits with $N = 2^m$.

For $|W_N\rangle$ with $N \neq 2^m$, we need to adjust p. For any block $B(p)$ applied on qubit b with control qubit a, p is the quotient of the number of "children" qubit of a (including itself) divided by the sum of a and b children (including a and b).

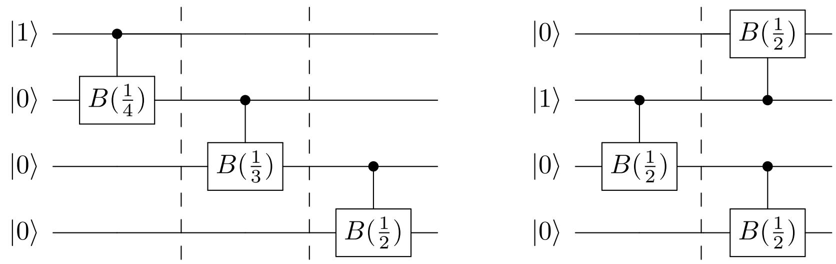
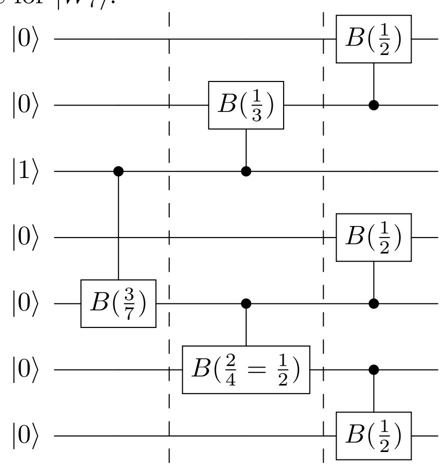


Figure 2: Circuits in $O(N)$ and $O(log_2(N))$ time-complexity for the creation of $|W_4\rangle$ states.

Here is an example for $|W_7\rangle$:



2.2 Tomography

A single-qubit measurement in the computational basis $\{|0\rangle, |1\rangle\}$ is a projection along the z-axis of the Bloch sphere. This single projection does not allow us to reconstruct the original state vector of the qubit. To do so, we need to perform 3 measurements along 3 linearly independent axes. The simplest approach is to do a first measurement in the computational basis, a second in the x basis and a third in the y basis. As IBMQ quantum devices allow measurement only in the computational basis, the first measurement is trivial, but the two others require additional manipulations. The idea is to apply a rotation U on the qubit in state $|\psi\rangle$ to get a new state $|\psi'\rangle = U|\psi\rangle$ such that its projection along z-axis is equivalent to a projection of $|\psi\rangle$ along the x-axis (or y-axis). The second measurement can by achieved by adding a Hadamard gate (H) to our circuit. This is equivalent to a rotation of π around the diagonal between the x- and z-axes on the Bloch sphere. To measure in the third basis, we use a $S^\dagger$ gate ($-\pi/2$ phase shift) followed by a Hadamard gate. Fig. 3 shows the measurement

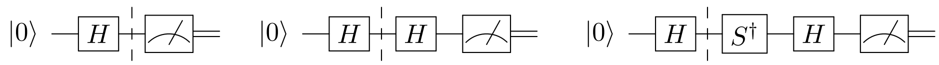


Figure 3: tomography circuits for $H|0\rangle$ circuit

of a single qubit state after a Hadamard gate

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}, \qquad S^\dagger = \begin{pmatrix}1 & 0\\ 0 & -i\end{pmatrix} = R_z\left(\frac{-\pi}{2}\right).$$

For multiple qubits, state vectors of each qubit are not independent. We cannot simply measure each one and aggregate the results. We need to test every measurement basis combination for all qubits, leading to 3^N different measures for $|W_N\rangle$. The Qiskit tomography module uses CVX [2], a disciplined convex optimisation tool, to compute the best quantum state that fits experimental results. The fidelity of the measured as compared to the theoretical one is given by $F(\rho_1, \rho_2) = Tr[\sqrt{\sqrt{\rho_1}.\rho_2.\sqrt{\rho_1}}]^2$, where ρ_1 and ρ_2 are density matrices for the theoretical and measured states, respectively [3].

2.3 Noise reduction

Real quantum computing devices are not ideal quantum computers, and even with regular calibrations experimental results are still impacted by noise. Overall, the coherence decays over time as $|1\rangle$ states tend to decay to $|0\rangle$. This is due to the physical implementation of qubits in IBMQ devices. Qubits are represented by electrons in a potential well. The lowest energy level corresponds to $|0\rangle$, while the next energy level is $|1\rangle$. The noise-reduction tool on Qiskit generates a series of calibration circuits (2^N circuits for N qubits, combinations of direct measure and NOT gate followed by measurements on every qubit). After executing them on a real device, experimental results are filtered using the results of calibration circuits.

3 Simulations

Before running a circuit on a real quantum computer, we can use the IBMQ quantum simulator. Although real quantum devices may run circuits faster than simulators, the overall time is shorter with the simulator in this project. Here, when using real quantum devices, we have to wait in a queue before our circuit can run. And even then, it takes a few seconds to load the circuit, and between each run the device performs a few calibration measurements. The overhead can go up to half a minute depending on the number of runs [4]. I was able to simulate a full tomography up to $|W_6\rangle$. With $N = 7$ (2187 different tomography circuits), my computer would run out of memory during the fitting phase (after using 15.5 GB of RAM memory plus 35 GB of swap memory). Figure 4 shows the state fidelity after tomography for $|W_N\rangle$ created with logarithmic-time circuits.

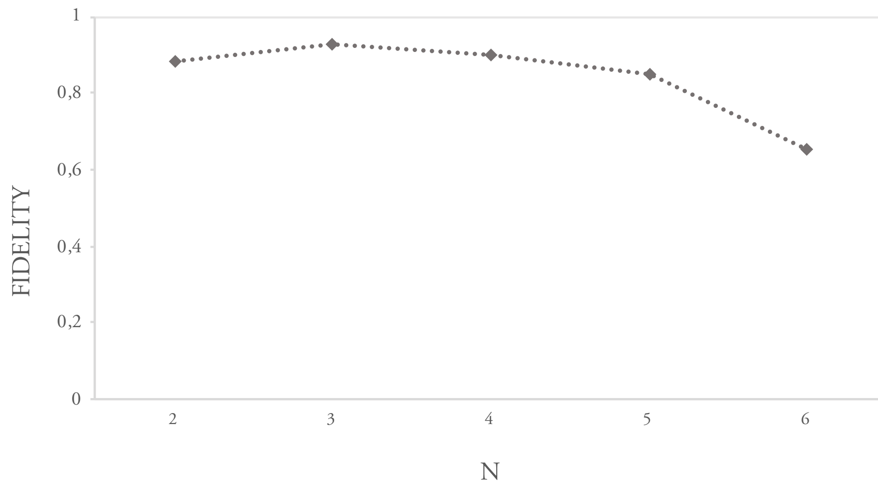


Figure 4: Fidelity of $|W_N\rangle$ states simulated with the logarithmic algorithm.

Every simulation is done with a noise model corresponding to the IBQ Q 16 Melbourne quantum device [5][6].

For $|W_6\rangle$, the drop in fidelity may come from the splitting of simulations when sent to the IBMQ simulator. Qiskit uses the HTTP protocol to communicate with the devices. With $N = 6$, we have $3^6 = 729$ tomography circuits, and the results are too large for a HTTP payload. We have to split the circuits in several chunks. As Qiskit does not support big jobs or the aggregation of multiple job results [7], I reassembled results manually. It is possible that the fitting algorithm used some metadata that I failed to aggregate properly. It is also possible that, considering the amount of data (roughly 10GB of memory usage for $|W_6\rangle$), the fitter chose a less greedy, but less efficient, optimisation method.

4 Experiments

All circuits ran on IBM Q 16 Melbourne, a publicly accessible 14 qubits quantum device [5]. Its topology allows to create up to $|W_8\rangle$ in 3 steps, and $|W_{12}\rangle$ in 4 steps. Experiments consist of two series: first with $O(N)$ time-complexity circuits, second with $O(log_2(N))$ time-complexity circuits. The goal was to use Qiskit tomography [8] and measurement-noise mitigation tools [6]. Figure 5 shows the fidelity of $|W_N\rangle$ states for $N = 2$ to 4, for both linear and logarithmic time-complexity circuits. For $N < 4$, operations cannot be performed in parallel, leading to the same circuit for linear and logarithmic methods. Results for $N = 4$ show a slight decrease in fidelity as compared to $N = 3$ in the logarithmic case, but a huge gap for the linear one. It demonstrates quite clearly the efficiency of the logarithmic algorithm. For $N = 5$, we see a drop in fidelity, maybe due to

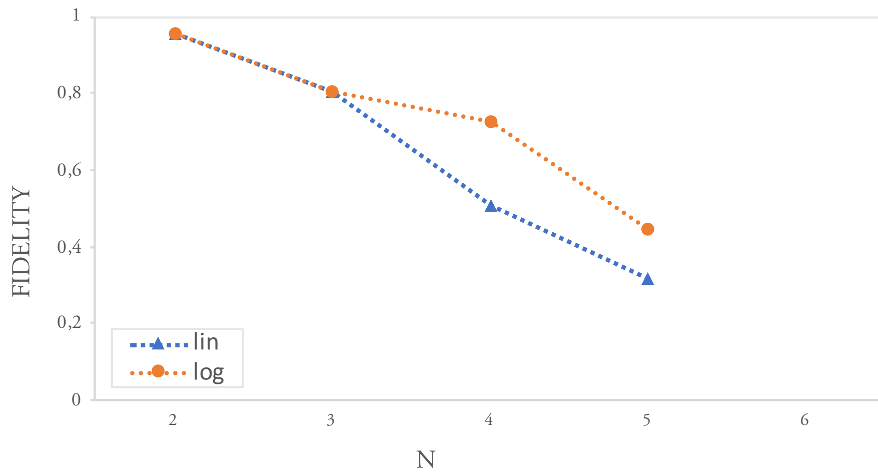


Figure 5: Fidelity of $|W_N\rangle$ states on a real quantum device.

limitations in the Qiskit API. A job can contain 27 circuits at most, and we can send at most 5 jobs in the queue. For $N = 5$, I had to split tomography circuits into 9 jobs (+1 for calibration measurements). It took almost 3 hours to get the second set of circuits running on the device. The noise may have changed in the device between the first set of measurements and my own calibrations. The fidelity decay could also come from a bad results aggregation, though the good fidelity observed at $N = 4$ seems to validate my aggregation method.

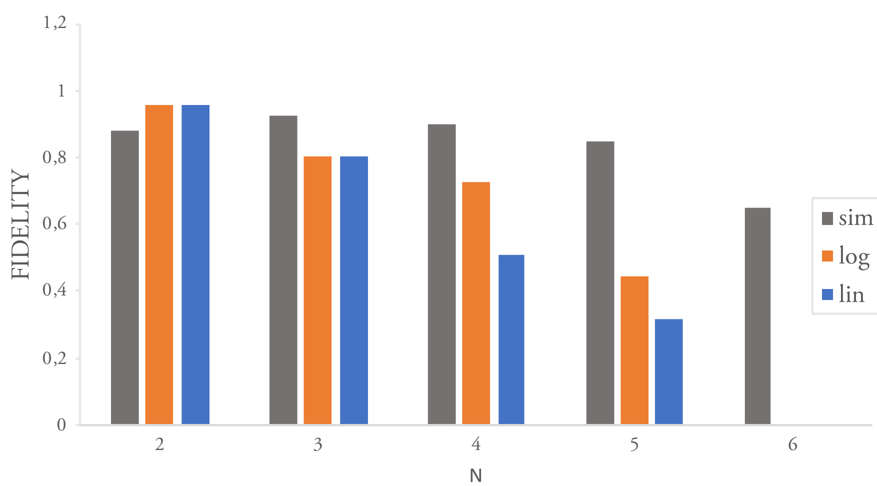


Figure 6: Comparison of simulated logarithmic, logarithmic, and linear circuits.

5 Conclusions

Tomography and noise mitigation tools from Qiskit API both proved to be quite powerful. The IBM quantum simulator showed a good ability for simulating ideal quantum circuits, but yielded too coherent results as compared to noisy outputs from quantum devices. The simulation of noise in quantum chips is not robust enough for large N. The main limitation for quantum tomography lies in the classical computing power to handle the output of real or simulated quantum circuits. For the moment, Qiskit is still in its beta version. The code is evolving quite fast. Functions, even whole classes, may break without deprecation warning. The communication between clients and IBMQ backends will surely evolve to address the payload limitation problem, and the issues regarding jobs with more than 27 circuits should be addressed in the next months.

References

[1] Diogo Cruz, Romain Fournier, Fabien Gremion, Alix Jeannerot, Kenichi Komagata, Tara Tosic, Jarla Thiesbrummel, Chun Lam Chan, Nicolas Macris, Marc-André Dupertuis, et al. Efficient quantum algorithms for ghz and w states, and implementation on the ibm quantum computer. *arXiv preprint arXiv:1807.05572*, 2018.

[2] CVX: Matlab software for disciplined convex programming. http://cvxr.com/cvx/. Accessed: 2019-06-07.

[3] Patrick J Coles, Stephan Eidenbenz, Scott Pakin, Adetokunbo Adedoyin, John Ambrosiano, Petr Anisimov, William Casper, Gopinath Chennupati, Carleton Coffrin, Hristo Djidjev, et al. Quantum algorithm implementations for beginners. *arXiv preprint arXiv:1804.03719*, 2018.

[4] Comparing run times on ibm quantum experience. https://quantumcomputing.stackexchange.com/a/3912. Accessed: 2019-06-07.

[5] IBM Q 16 Melbourne backend specifications. https://github.com/Qiskit/ibmq-device-information/tree/master/backends/melbourne/V1. Accessed: 2019-06-12.

[6] Qiskit documentation on noise simulation. https://qiskit.org/documentation/aer/device$_noise_simulation.html. Accessed: 2019-06-07$.

[7] A proper way to combine the result object. https://github.com/Qiskit/qiskit-terra/issues/1470. Accessed: 2019-06-05.

[8] Qiskit documentation on tomography. https://qiskit.org/documentation/ignis/tomography.html. Accessed: 2019-06-07.