Implementing a quantum simulation algorithm with Qiskit

Floris van den Ende Sandy Bridgwater Antonio Mendes
Terts Diepraam

May 29, 2020

The aim of this project was to implement the Spectrum by Quantum Walk algorithm for the quantum chemistry problem as described by Poulin et al. [3], specifically for the hydrogen atom. The conventional method for quantum simulation is to approximate the unitary evolution operator $U = \exp\{-iHt\}$. This algorithm circumvents having to approximate it, as the algorithm makes use of a unitary operator which is a function of the Hamiltonian. As a consequence, the energy of the system is encoded in the phase of that operator, so we can perform phase estimation to obtain the energy of the system. The operator is defined as

$$W = SVe^{i\pi} \tag{1}$$

Where S and V are defined as following. We also define $|\beta\rangle$, which is the state we need to initialize the auxillary register to [3].

$$S = (B(I - 2\left|0\right\rangle\left\langle 0\right|)B^{\dagger}) \otimes I = (I - 2|\beta\rangle\langle\beta|) \otimes I)$$

$$V = \sum_j \left|j\right\rangle\left\langle j\right| \otimes P_j$$

$$|\beta\rangle = B\left|0\right\rangle = \sum_j \beta_j \left|j\right\rangle$$

Here β_j and P_j are terms of the Hamiltonian, which after Jordan-Wigner encoding takes the form

$$\bar{H} = \frac{H}{\mathcal{N}} = \sum_j |\beta_j|^2 P_j. \tag{2}$$

As we aim to simulate the hydrogen molecule, we take the specific Hamiltonian of the hydrogen molecule which is:

$$\begin{aligned}\hat{H}_{BK} = &-0.81261I + 0.171201\sigma_0^z + 0.16862325\sigma_1^z - 0.2227965\sigma_2^z + 0.171201\sigma_1^z\sigma_0^z \\ &+ 0.12054625\sigma_2^z\sigma_0^z + 0.17434925\sigma_3^z\sigma_1^z + 0.04532175\sigma_2^x\sigma_1^z\sigma_0^x + 0.04532175\sigma_2^y\sigma_1^z\sigma_0^y \\ &+ 0.165868\sigma_2^z\sigma_1^z\sigma_0^z + 0.12054625\sigma_3^z\sigma_2^z\sigma_0^z - 0.2227965\sigma_3^z\sigma_2^z\sigma_1^z \\ &+ 0.04532175\sigma_3^z\sigma_2^x\sigma_1^z\sigma_0^x + 0.04532175\sigma_3^z\sigma_2^y\sigma_1^z\sigma_0^y + 0.165868\sigma_3^z\sigma_2^z\sigma_1^z\sigma_0^z\end{aligned}$$

Technically, we'd need a $4^4 = 256$-dimensional matrix to express this Hamiltonian. However, as there are only 16 terms, we can express this with four qubits, as we can change the basis to only represent those 16 terms. This means that each β_j term corresponds to the $|j\rangle$ bitstring state.

So first, we define three registers: `counting`, `aux` and `main`.

We initialize the `aux` register to the state $|\beta\rangle$. We do so by using the `initialize` function of Qiskit, which takes an n-dimensional vector (in our case, the list of β_j without their corresponding pauli matrices) and initializes the $\log n$ qubits in the state corresponding to that vector.

Next, we have to put the `main` register in the groundstate of the Hamiltonian. Usually for larger systems this is a laborious process as it requires a lot of computing power to calculate, due to the exponantially growing Hilbert space. However, hydrogen is a simple system and the groundstate has long been calculated classicaly, via the Hartree-Fock method. We prepare this register in the groundstate by using the `HartreeFock` function of the chemistry module of qiskit.

Then, we have to create the controlled-W operation eq. (1) by combining the S and V operations. Initially, we thought that the S operator had the same gate operations as the Grover Diffusion operator, found in [1]. In that case, we could easily extrapolate the three-qubit example they gave to a four-qubit operation, by creating a four-qubit Toffoli gate with an extra ancillary qubit. Soon we realized that that is not the actual representation of S, as the operators are different in nature. We haven't been able to correctly implement S.

Similarly, we thought that the V operation was just a series of tensor products of the pauli matrices from the Hamiltonian. This is also not the case, as V should be a controlled operation. We don't know how to implement V either.

The next step is performing phase estimation of W. Since we hadn't been able to implement W, we performed phase estimation of a random unitary. The code for phase estimation was adapted from [1], which is in turn based on Nielsen and Chuang [2]. The main complication in this part was that the original code used 3 counting qubits, instead of 2. Therefore, we adapted it to support any number of qubits. Additionally, the bits had to be adapted to fit the registry structure. After a series of controlled-U operations, we apply inverse fourier transform to the `counting` register. Then the state of the counting qubits should be $|\varphi\rangle$, where φ is the phase of U. Then performing measurement on the counting qubits, projected on a register of classical qubits, gives us the phase and hence the energy. A schematic of the circuit we implemented can be found in fig. 1.

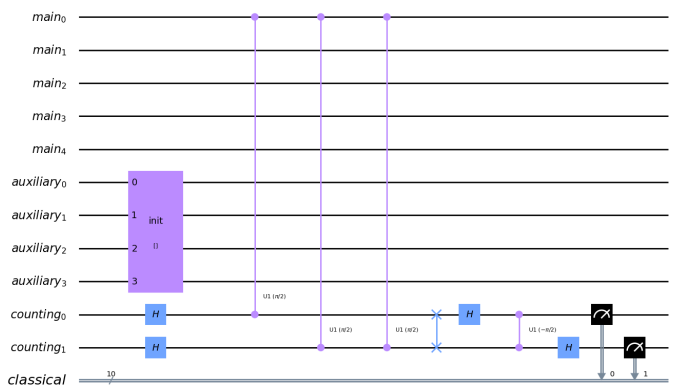

Figure 1: The full circuit.

To run the code on a quantum computer, the circuit was significantly simplified in order to run it on 3 qubits. The result was a circuit that performs a phase estimation with a single qubit in the `main` register and 2 qubits in the `counting` register. After creating an account and generating an API token, we used the following code in listing 1 to run it on an IBM-Q computer. The results can be found in fig. 3. The angle that the operation in the circuit applies was smaller than our resolution, so this result was expected. All the code can be found in `https://github.com/FlorisvdEndeo/qiskit`.

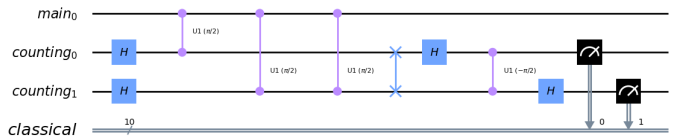

Figure 2: The simplified circuit.

```
IBMQ.load_account()
provider = IBMQ.get_provider(hub='ibm-q')
provider.backends()
backend = ibmq.least_busy(provider.backends(
    filters=lambda b: b.configuration().n_qubits >= 3
                      and not b.configuration().simulator
                      and b.status().operational==True))
job = execute(circ, backend=backend, shots=8000)
result = job.result()
```

Listing 1: Code for running on the IBM computer

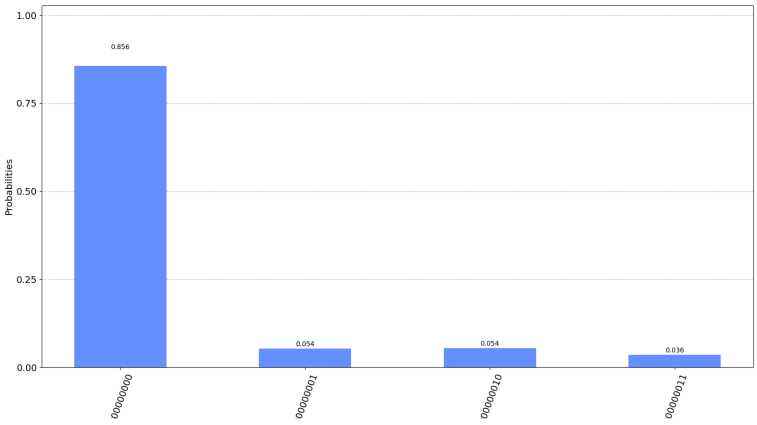

Figure 3: Results of the simplified algorithm.

References

[1] A. Asfaw et al. "Quantum Phase Estimation". In: *Learn Quantum Computation Using Qiskit*. 2020. URL: https://qiskit.org/textbook/ch-algorithms/quantum-phase-estimation.html.

[2] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. 10th. USA: Cambridge University Press, 2011. ISBN: 1107002176.

[3] D. Poulin et al. "Quantum Algorithm for Spectral Measurement with a Lower Gate Count". In: *Physical Review Letters* 121.1 (July 2018). ISSN: 1079-7114. DOI: 10.1103/physrevlett.121.010501. URL: http://dx.doi.org/10.1103/PhysRevLett.121.010501.