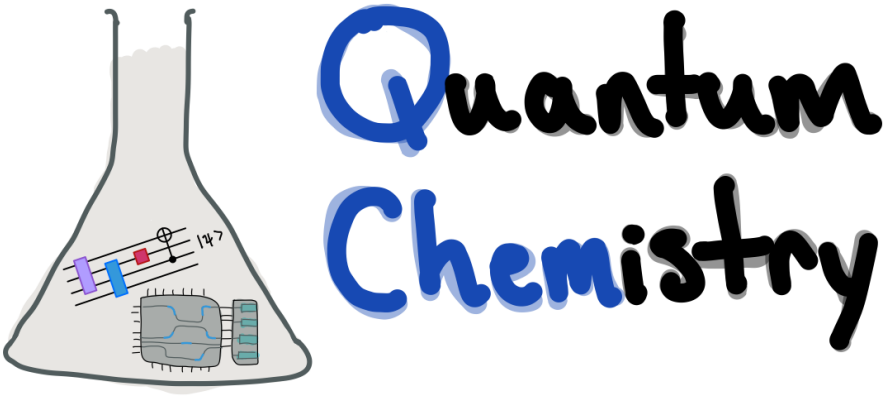

Qchem: Optimizing Measurements [200 points]

Version: 1

Quantum Chemistry

Numerical techniques for determining the structure and chemical properties of molecules are a juggernaut area of research in the physical sciences. *Ab initio* methods like density functional theory have been a staple method in this field for decades, but have been limited in scalability and accuracy. As such, chemistry applications and problems are desirable candidates for demonstrating a quantum advantage.

It is therefore no surprise that quantum chemistry is one of the leading application areas of quantum computers. In the **Quantum Chemistry** category, you will be using PennyLane's core quantum chemistry functionalities to become familiarized with concepts and tools developed in this sub-field of quantum computing, like mapping molecular Hamiltonians to qubit Hamiltonians and quantum gates that preserve the electron number. Beyond these five questions in this category, there is a plethora of informative tutorials on the PennyLane website that will boost your understanding of topics that we will cover in this category. Let's get started!

Problem statement [200 points]

One of the fundamental algorithms to determine the electronic structure of molecules is the Variational Quantum Eigensolver (VQE). To carry out this algorithm we must know the energy of the molecule given an electronic configuration or structure.

This structure is encoded in a quantum circuit with a set of initial gates and, to

calculate the energy of that configuration, we must obtain the expected value with respect to the Hamiltonian defined by the molecule.

However, given an arbitrary Hamiltonian, we cannot calculate this value directly so we usually decompose it into a linear combination of Pauli operators. In this way, we can calculate the expected value of each of these operators and reconstruct the energy of the Hamiltonian.

A Pauli operator is defined by the tensor product of the operators X, Y and Z applied on different qubits. Suppose for example that the configuration is defined by a circuit of 4 qubits and we are interested in calculating the expected value with respect to the operator $Z(0) \otimes Y(1)$ (where we are indicating in parentheses the qubit where we want to execute this operator). Then what we must do is perform a series of executions of the circuit, measuring in the Z axis on qubit 0 and on the Y axis on qubit 1. In this case, we would not need to perform any measurements on the other qubits.

Let's look at the following example:

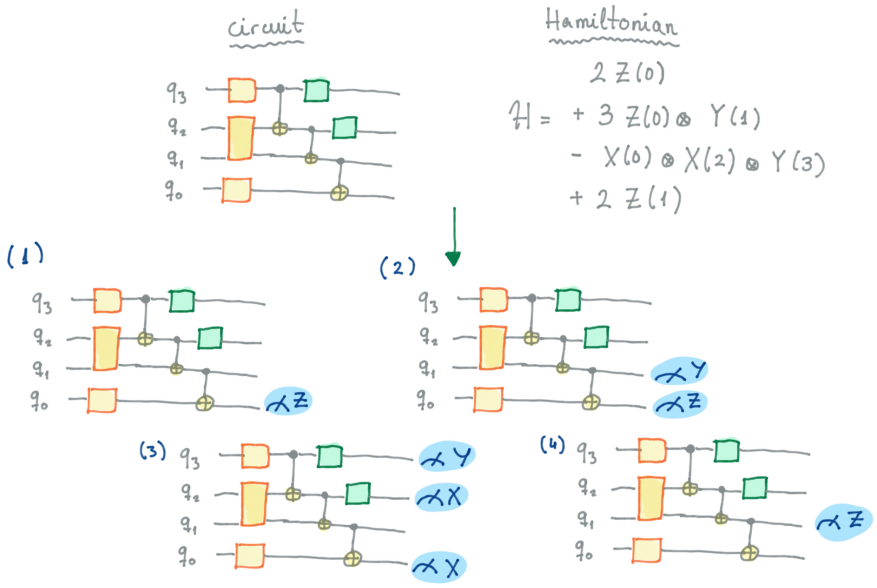

Figure 1: Decomposition

In this case we have a Hamiltonian that we have divided into 4 Pauli operators so, given an electronic configuration (defined by the initial circuit), we will have to run 4 different circuits: 1 for each operator of the decomposition. The number of circuits required in a general Hamiltonian grows like $O(N^4)$, with N being the number of qubits, so it is desirable to look for techniques that reduce this number. To address this problem, several techniques have been developed so in

this challenge we will work with a very simple one. There are two rules we can play with:

- If two circuits do not intersect in any of their measurement qubits, they could be executed at the same time. For example:

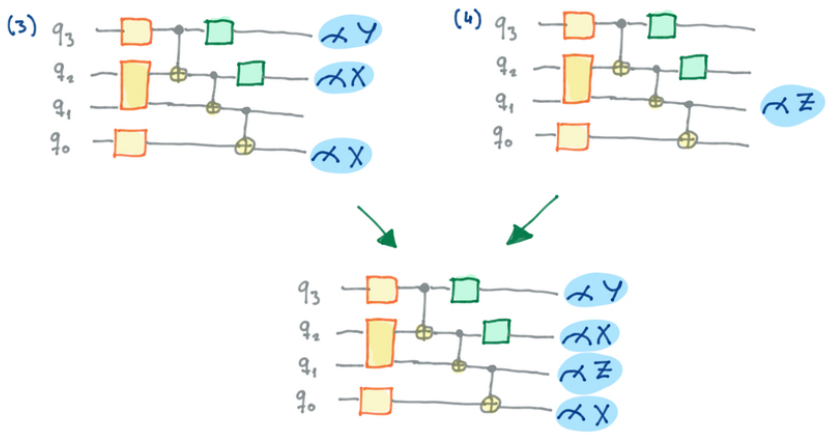

Figure 2: Optimization 1

With this, by processing the output of the final circuit we can obtain the equivalent results to the two initial circuits, thus reducing the number of executions required.

- If two circuits intersect in some of their measurement qubits but the measurement operators also coincide in all of them, they can be joined. Example:

In this problem, you will be tasked with compressing the number of measurements needed for some given Hamiltonians, using the above observations. During the whole problem we will focus only on the observables so we will not worry about the coefficients that form the Hamiltonian. ### Input

- `obs_hamiltonian` (list(list(str))): Groups of Pauli operators making up the Hamiltonian. Each row in the `obs_hamiltonian` corresponds to a single term in the Hamiltonian. For example:

```
obs_hamiltonian = [
    ["Z", "I", "I", "I"],
    ["Z", "Y", "I", "I"],
    ["X", "I", "X", "Y"],
    ["I", "Z", "I", "I"]
]
```

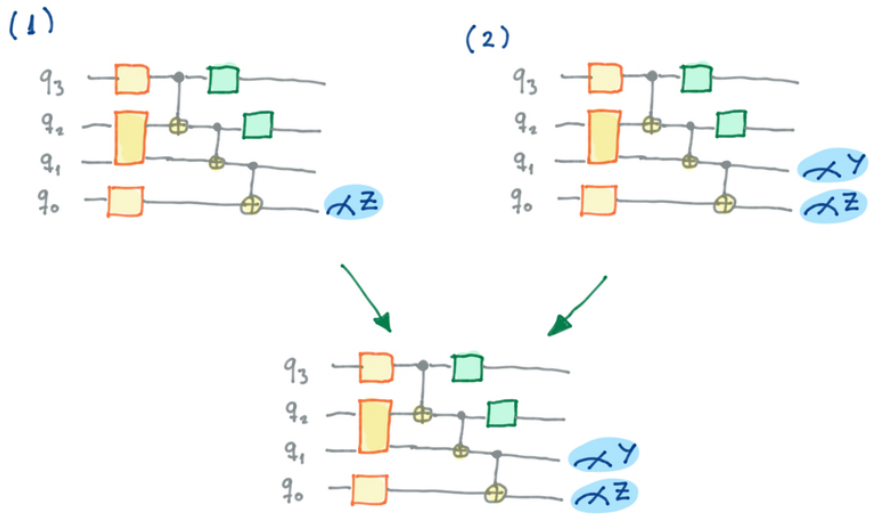

Figure 3: Optimization 2

The element "I" (Identity), will be placed in the positions where we will not make any measurement. The number of qubits and terms are arbitrary.

Output

The output of `optimize_measurements` should be a list with the same format but in which we have reduced the number of terms to be executed. In this case it would be:

```
output = [
    ["Z", "Y", "I", "I"],
    ["X", "Z", "X", "Y"],
]
```

- list(list(str)): the chosen Pauli operators to measure after grouping.

Finally, the compression ratio (r) of the algorithm will be taken to evaluate the solution:

$$r = 1 - \frac{\text{nº final observables}}{\text{nº initial observables}}$$

Acceptance Criteria

In order for your submission to be judged as "correct":

- The outputs generated by your solution when run with a given `.in` file must match those in the corresponding `.ans` file to within the `1.0e-4` tolerance specified below.

- Your solution must take no longer than the `60s` specified below to produce its outputs.

WARNING: Don't modify any of the code in the template outside of the `# QHACK #` markers, as this code is needed to test your solution. Do not add any print statements to your solution, as this will cause your submission to fail.

Specs
Tolerance: **0.0001**
Time limit: **60 s**

Version History

Version 1: Initial document.