Hi!

Once again, thank you for your interest in the QC Mentorship program!

We decided to select participants based on how they will manage to do some simple “screening tasks”.

These tasks have been designed to:
- find out if you have the skills necessary to succeed in our program.
- be doable with basic QC knowledge - nothing should be too hard for you to quickly learn.
- allow you to learn some interesting concepts of QC.
- give you some choices depending on your interests.

What we mean by skills is not knowledge and expertise in QC. It’s the ability to code, learn new concepts and to meet deadlines.

What are we looking for in these applications?
- Coding skills – clear, readable, well-structured code
- Communication – well-described results, easy to understand, tidy.
- Reliability – submitted on time, all the points from the task description are met
- Research skills – asking good questions and answering them methodically

Also, feel free to be creative – once you finish the basic version of the task, you can expand it.

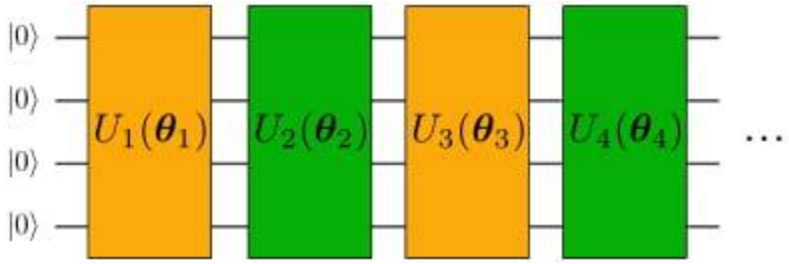
Choose tasks based on your interests, don’t try to pick the easiest one.

<u>You need to do only 1 task.</u> Feel free to do all of them, it might be a good learning opportunity, but it won’t affect admissions to the program :)

So here are the tasks:

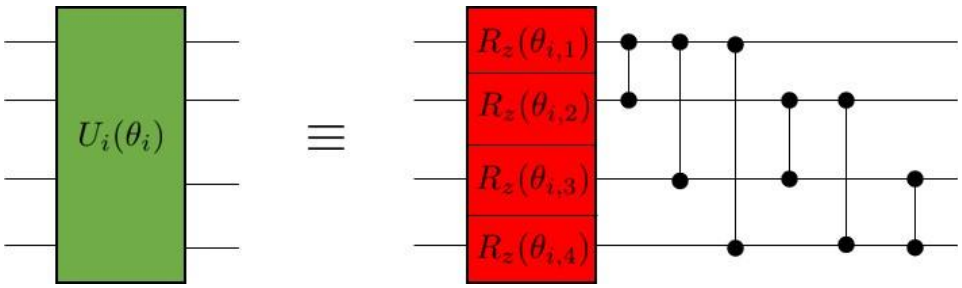
Task 1

Implement, on a quantum simulator of your choice, the following 4 qubits state $|\psi(\theta)>$:

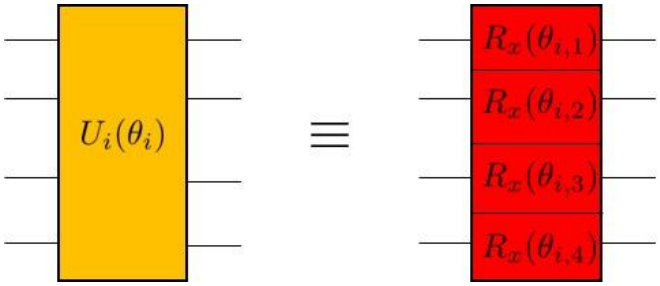


Where the number of layers, denoted with L, has to be considered as a parameter. We call ¨Layer¨ the combination of 1 yellow + 1 green block, so, for example, U1 + U2 is a layer. The odd/even variational blocks are given by:

Even blocks



Odd blocks



The angles $\theta_{i,n}$ are variational parameters, lying in the interval $(0,\ 2\pi)$, initialized at random.
Double qubit gates are CZ gates.

Report with a plot, as a function of the number of layers, L, the minimum distance

$\epsilon = min_{\theta} \,||\, |\psi(\theta)> \; - \; |\phi> \,||$

Where $|\phi>$ is a randomly generated vector on 4 qubits and the norm $||\,|v>\,||$, of a state $|v>$, simply denotes the square root of the sum of the modulus square of the components of $|v>$.
The right set of parameters $\theta_{i,n}$ can be found via any method of choice (e.g. grid-search or gradient descent)

Bonus question:
Try using other gates for the parametrized gates and see what happens.

Task 2

Implement a circuit that returns |01> and |10> with equal probability (50% for each).
Requirements :

- The circuit should consist only of CNOTs, RXs and RYs.
- Start from all parameters in parametric gates being equal to 0 or randomly chosen.

- You should find the right set of parameters using gradient descent (you can use more advanced optimization methods if you like).
- Simulations must be done with sampling (i.e. a limited number of measurements per iteration) and noise.

Compare the results for different numbers of measurements: 1, 10, 100, 1000.

Bonus question:
How to make sure you produce state |01> + |10> and not |01> - |10> ?

(Actually for more careful readers, the “correct” version of this question is posted below:
How to make sure you produce state |01⟩ + |10⟩ and not any other combination of |01> + e(i*phi)|10⟩ (for example |01⟩ - |10⟩)?)

Task 3

Please write a simple compiler – program, which translates one quantum circuit into another, using a restricted set of gates.

You need to consider just the basic gates for the input circuit, such as (I, H, X, Y, Z, RX, RY, RZ, CNOT, CZ).

The output circuit should consist only from the following gates: RX, RZ, CZ. In other words, each gate in the original circuit must be replaced by an equivalent combination of gates coming from the restricted set (RX, RZ, CZ) only.

For example, a Hadamard gate after compilation looks like this:

RZ(pi/2)
RX(pi/2)
RZ(pi/2)

Analyze what’s the overhead of the compiled program compared to the original one and propose how to improve it. What we mean by overhead is the following: by replacing all the initial gates with the restricted set of gates given in the problem, you will see that the resulting circuit is much more involved than the original one. This is what we called the overhead, and you may think about how to treat this problem, i.e. you could try to simplify as much as possible the resulting circuit.

Task 4

Find the lowest eigenvalue of the following matrix:

[1 0 0 0;
0 0 -1 0;
0 -1 0 0;
0 0 0 1]

using VQE-like circuits, created by yourself from scratch.

In general, this exercise might be pretty difficult, so below there are a couple of tips. We've written them in a pale font so that those of you who embrace the challenge don't look at them accidentally ;)
PLEASE DON'T CHANGE THE COLOR OF THE FOLLOWING PARAGRAPH!
OTHER PEOPLE SEE THE CHANGES – IF YOU WANT TO READ WHAT'S THERE, JUST COPY IT ELSEWHERE.

1. It requires decomposing the matrix to the sum of Pauli terms.
2. Decomposition involves only terms consisting of the same matrices, i.e.: II, XX, YY, ZZ, and the coefficients are from the set [-1/2, -1, 0, 1, 1/2].
3. The ansatz you can use is: (RX I) CX (HI) |00>, where angle in RX is your variational parameter.
4. You can find an explanation of VQE in this blog post. You can also find links to further resources there.
5. You can just search through all angles for RX, you don't need to use any optimizers like gradient descent.

Deadline
2 weeks from when you've submitted your application in your timezone.

Once you have finished a screening task, please submit your GitHub repository containing the code to this google form: https://forms.gle/2S6Z912fzExLhg5aA -- other forms of submission will not be accepted!

If you have any questions - please add comments to this document, or ask it in the QOSF slack channel (the link to join should be in the first follow-up email sent upon the submission of your application). We will be updating this document with more details and/FAQ to avoid confusion, so make sure to check it before asking :)

Have a nice day!
Michał Stęchły

FAQ

Q: Can we use any quantum libraries or are we restricted to a particular set of tools?

A: Feel free to use whatever you like, just make sure that the tool doesn't solve the whole problem for you.
Regarding the language of choice, Python is definitely the preferred one, since this is the language that most of the mentors use.
You can do the task first in the language of your preference and then translate it to Python if that's more convenient for you.

Q: I am applying as a member of a team. How many tasks do we submit?

A: Each member of a team must submit their own screening task. This will help us judge the skill level of each individual team member and help us pair folks up with the right mentor.

Q: How should I submit the solution?

A: All the materials for the submission should be inside the GitHub repository. Please do not send us any loose files as attachments or in any other format. Please submit your GitHub repository to this google form once you've finished: https://forms.gle/2S6Z912fzExLhg5aA

Q: My team-mate wants to leave the team because she can't manage these along with exams. So will this affect our team status or anything like that?

A: Well, just let us know and you can continue as an individual/smaller team.