CLASSIQ

Classiq's Challenge for YQuantum 2024: Sparse State-Preparation

---

Welcome to Classiq's Sparse State-Preparation Challenge, for |Y⟩Quantum 2024. This document serves as a comprehensive challenge guide, designed for quantum computing enthusiasts at all levels. From beginners to seasoned experts, we aim to enrich your understanding of sparse state-preparation and spur innovation in the field of quantum computing, specifically focusing on the nuanced domain of sparse state-preparation. Classiq's platform, QMOD language, and SDK offer powerful abstractions for expediting the process of quantum algorithm design, prototyping, implementation, and testing. Your team will leverage these abstractions in order to develop a competitive solution within a 24 hour period.

Background: State Preparation

State preparation underpins quantum computing, acting as the foundational step for a multitude of quantum algorithms across diverse fields such as quantum linear algebra, quantum chemical modeling, quantum search protocols, and quantum machine learning. It involves configuring a quantum state to represent a specific vector of probabilities or a predefined function. The efficiency of state preparation is a critical determinant in harnessing the true potential of quantum computational advantages, particularly in scenarios where classical computation becomes impractical due to exponential increases in resource requirements. Classiq's SDK and QMOD language have built-in functions for quantum state preparation. However, it is not computationally efficient for sparse state preparation.

The Problem: Sparse State-Preparation

Sparse state-preparation, a specialized subset of state preparation, emphasizes the generation of quantum states characterized by a significant proportion of zero amplitudes. This feature is prevalent in numerous quantum computing applications, where the desired quantum state comprises a small number of non-zero amplitudes amidst a potentially vast set. By exploiting the inherent sparsity, sparse state-preparation algorithms aim to drastically reduce the quantum resources required, such as quantum gates and circuit complexity. This aspect is vital for the practical execution of complex quantum algorithms on near-term quantum hardware, which is constrained by limitations in qubit count, gate fidelity, and coherence times.

Challenge Brief

Objective: The challenge is designed to inspire participants to develop quantum circuits capable of efficiently preparing sparse quantum states. Entrants are tasked with creating circuits that load a quantum state with probabilities corresponding to a given sparse vector of **probabilities**, ensuring the presence of

only a **constant number of non-zero probabilities**. The key lies in achieving this with minimal resource utilization and maximal fidelity. "An Efficient Algorithm for Sparse Quantum State Preparation" by Niels Gleinig and Torsten Hoefler provides a two-part algorithm for sparse state preparation. Participants should read, understand, and base their implementation off of the paper.

Given a list of indices and their respective probabilities, your team is tasked with creating a circuit such that it loads a state with amplitudes in the magnitudes of these probabilities. The circuit will not take phase amplitude into account, meaning that a_i will always be a square root of a real number.

$$|\psi\rangle_n = \Sigma_i a_i |i\rangle_n \text{ such that } |a_i|^2 = p_i .$$

(notice that your team has the freedom to load the amplitude with an arbitrary phase).

Example Scenario:

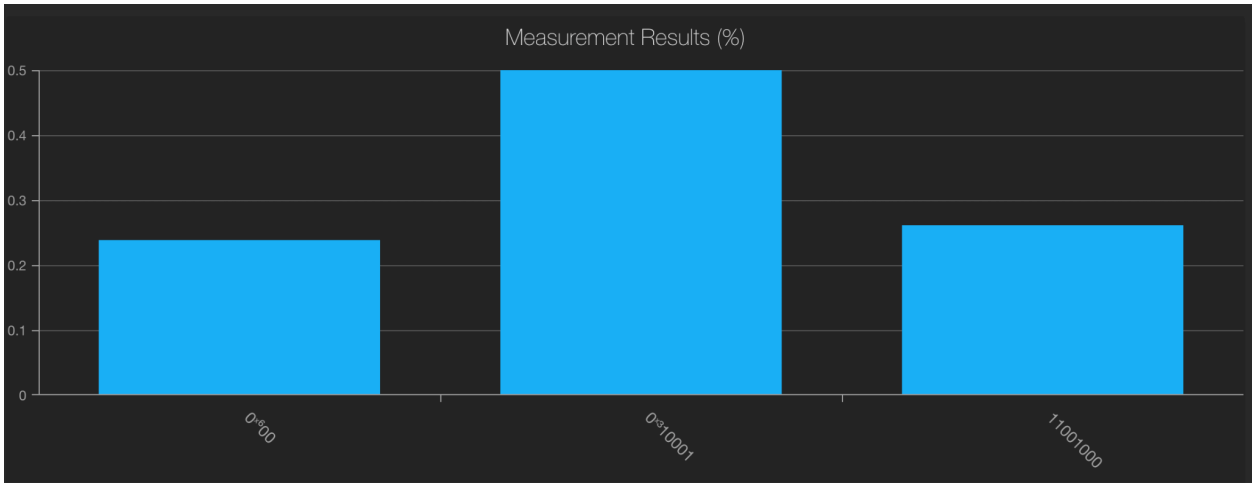
- Input: A probabilities dictionary, such as:

Sparse_states = {'00000001': 0.25, '00010001': 0.5, '11001000': 0.25}

- Output Goal: Efficiently load a quantum state reflecting these probabilities, such as:

$$|\psi\rangle_8 = \sqrt{0.25}|1\rangle_8 + \sqrt{0.5}|17\rangle_8 + \sqrt{0.25}|200\rangle_8$$

- A sample output for the specified example:



Detailed Considerations for Participants

- Auxiliary Qubits: Their use is permitted, provided they do not remain entangled or in non-zero states post-operation.
- Adaptability and Efficiency: Solutions should be versatile, considering the trade-offs between circuit depth, number of auxiliary qubits, gate count, and other circuit parameters.
- Innovative Approach: Encourage exploration beyond conventional methodologies.

Comprehensive Submission Guidelines

Participants are required to submit:

1. `.qmod` and `.qprog` files showcasing their quantum circuit design and implementation.
2. Results demonstrating the execution of their circuit, saved as JSON.
3. The Python code employed for model generation.
4. An in-depth explanation detailing their approach and methodology. Create a README, outlining an abstract overview of your team's implementation. The README should outline the functional outline of all Quantum Functions used in your program, including their parameters/arguments and outputs.

Evaluation Criteria

Submissions will be assessed based on:

1.) Target Problem Solving: Correct sparse quantum state preparation for the following scenarios:

- For 6 qubits: Probs = {‘000011’: 0.4, ‘111111’: 0.6}
- For 10 qubits: Probs = {‘0000000101’: 0.33, ‘0000010001’: 0.33, ‘0001100010’: 0.34}
- For 20 qubits: Probs = {‘00000000000000000101’: 0.6, ‘00000000000000010001’: 0.4}

2.) Optimization Balance: Utilizing Pareto analysis to evaluate the balance between circuit depth and auxiliary qubit usage.

3.) Originality and Creativity: Encouraging unique and innovative solutions.

4.) Scalability and Versatility: The ability to efficiently manage different numbers of qubits and non-zero coefficients.

Guidelines for Implementation

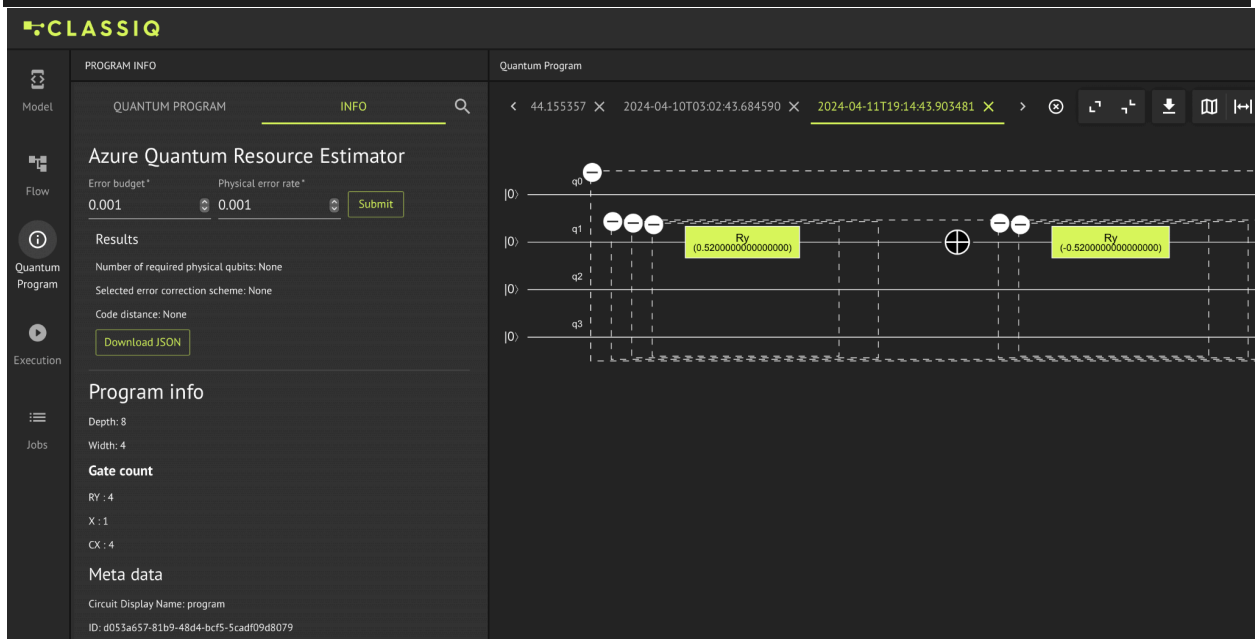
- Foundational Understanding: Solidify your grasp of quantum state preparation fundamentals and the critical role of sparsity.
- Resource Optimization: Prioritize minimizing quantum resources while maximizing performance and accuracy.
- Circuit Design Exploration: Engage with various quantum circuit design and optimization strategies to discover the most efficient approach for sparse state preparation.
- Rigorous Testing: Ensure thorough testing and validation of your circuits to confirm the accurate preparation of the desired sparse quantum states.

Keep Note

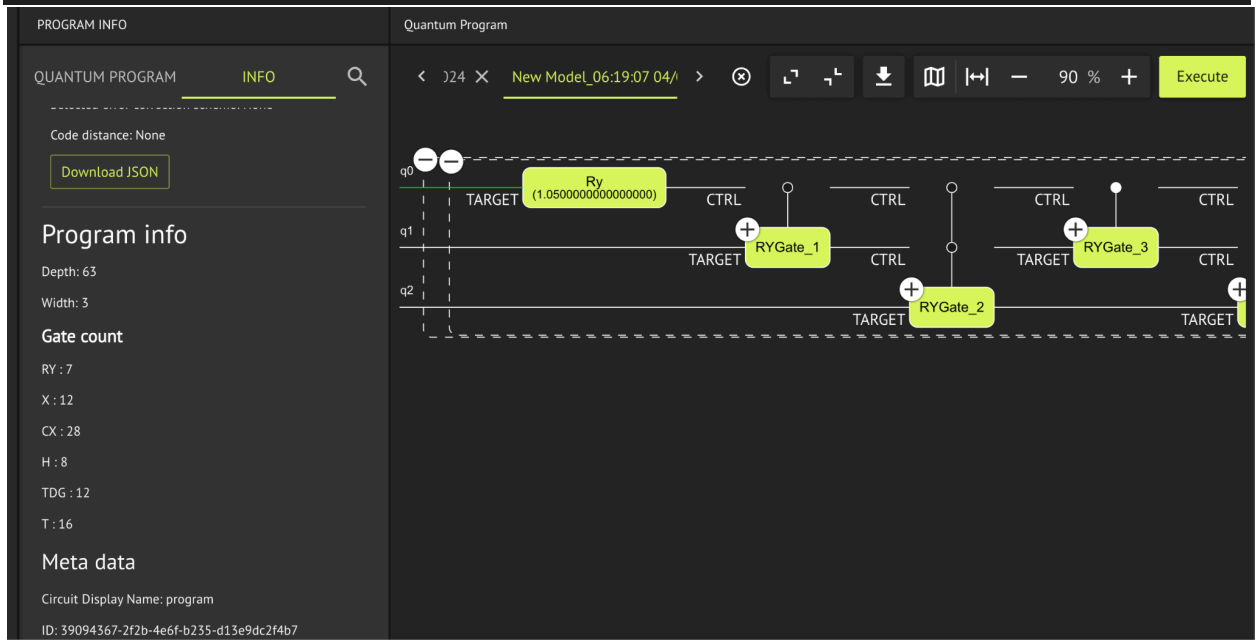
You're tasked with creating quantum circuits to prepare specific states. You are permitted to utilize auxiliary qubits. Nonetheless, it's essential to ensure that at the circuit's conclusion, there are no residual non-zero or entangled qubits apart from the designated loaded state. You will be provided a template notebook with some already defined functions. You are tasked with implementing the remaining logic based on the two-part algorithm described in the provided research paper, "An Efficient Algorithm for Sparse Quantum State Preparation".

There's a spectrum of potential solutions for this task. The efficiency of particular approaches might fluctuate based on problem dimensions. Furthermore, one must navigate trade-offs between multiple circuit parameters: the circuit's depth, the count of auxiliary qubits, gate numbers, and so on. Take note of the following example comparing *prepare_sparse_states* and the built-in *prepare_state*. Notice that a proper implementation of *prepare_sparse_states* creates a circuit with depth of 8. In comparison the built-in *prepare_state* function creates the same quantum state with a circuit depth of 63.

```
@qfunc
def main(psi: Output[QArray[QBit]]):
    sparse_state_dic ={'000':0.25, '001':0.5, '111':0.25}
    prepare_sparse_state(dictionary=sparse_state_dic,out=psi)
```



```
@qfunc
def main(psi: Output[QArray[QBit]]):
    sparse_state_dic ={'000':0.25, '001':0.5, '111':0.25}
    sparse_state_array = convert_dic2array(sparse_state_dic)
    prepare_state(sparse_state_array,0.001,psi)
```



Best of Luck to All Participants!

Participating in this challenge presents a unique opportunity to contribute to the advancement of quantum computing technology and to explore new frontiers in the quantum domain. We invite you to leverage your creativity, knowledge, and passion for quantum computing to tackle this engaging and challenging problem.