CREATING A STATE-OF-THE-ART GPU-BASED SIMULATOR FOR QUANTUM ALGORITHMS AND ERROR-CORRECTION

University of Victoria

Dominic Largoza
Department of Physics & Astronomy

Supervised by Thomas E. Baker
Departments of Physics & Astronomy and Chemistry

March 8th, 2026

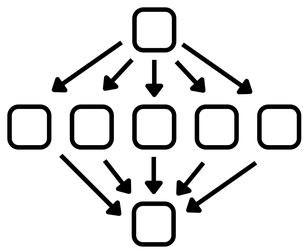
Introduction

Quantum computing utilizes phenomena in quantum mechanics to develop new algorithms to perform calculations and measurements.

Unlike classical computers, which use classical bits that measure 0 or 1, quantum computers use qubits, which allow states to be in a superposition of 0 and 1.

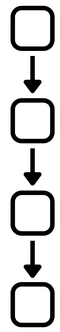
These quantum algorithms can potentially be faster than their classical counterparts, but need further development.

GPU

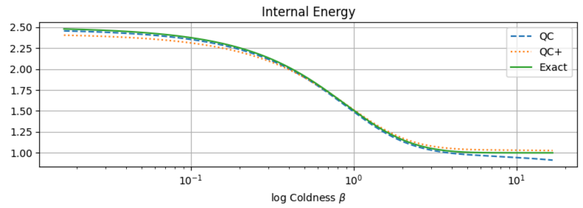


GPU's have many cores compared to CPU's which allows them to parallelize tasks. Meaning that tasks can be broken up into pieces and then performed simulatneously.

CPU



Results



Simple example simulation to measure the internal energy in a Green's function system.

Quantum Error Correction

A downside to quantum computing however is that qubits are very noisy, meaning they are highly prone to error. Without being able to correct or mitigate these errors, quantum computing would be futile.

The upper critical threshold of a quantum error correcting code can be found by mapping the quantum system to a classical system. The classical system can then be simulated using GPU's through various methods such as Monte Carlo simulation.

Monte Carlo simulation utilizes random number generation to simulate a system or estimate probability densities. For example, we used the Metropolis-Hastings algorithm to simulate the classical model needed to determine the upper critical threshold.

Objective

To create a simulator that utilizes GPU's to develop quantum algorithms and analyze quantum error correcting codes.

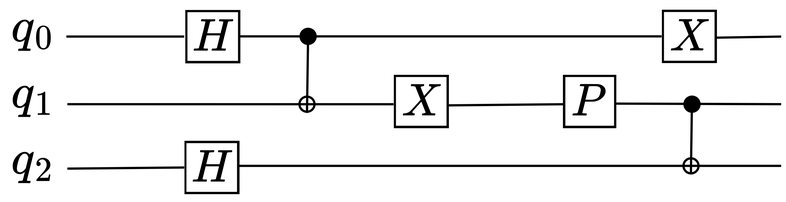
Quantum Gates

Quantum computing uses quantum gates, which are analogous to logic gates in classical computing. Which can be represented as matrices.

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad P(\theta) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{bmatrix}$$

Quantum Circuits



Methodology

1. Designed and developed the quantum circuit or algorithm in Qiskit. Then decomposed the circuit into base gates.
2. Deciphered the quantum circuit using OpenQASM.
3. Translated the output from OpenQASM to CUDA-Q and simulated the circuit using GPU's from NVIDIA.

Conclusion

GPU simulation can be implemented in the advancement of quantum computing from the development of quantum algorithms to analyzing quantum error correcting codes. Allowing researchers to calculate and design various systems easier.

Key Sources & Acknowledgements

International Business Machines Corporation. (n.d.). IBM Quantum. IBM. https://www.ibm.com/quantum
NVIDIA Corporation. (n.d.). CUDA-Q. NVIDIA Developer. https://developer.nvidia.com/cuda-q
Dennis, E., Kitaev, A., Landahl, A., & Preskill, J. (2001). Topological quantum memory (arXiv:quant-ph/0110143). arXiv. https://arxiv.org/abs/quant-ph/0110143
Nielsen, M. A., & Chuang, I. L. (2010). Quantum computation and quantum information (10th anniversary ed.). Cambridge University Press.

This research was supported by the Jamie Cassels Undergraduate Research Awards, University of Victoria.
This work has been supported in part by the Mitacs Accelerate Umbrella program.
A special thanks to Luca Adams for the help and development of the results for this project.

mitacs