Measurement-Based Quantum Computing

Utku Birkan
Advisor: Sadi Turgut

Department of Physics, ODTÜ

Abstract

In this prjoect we insvestigate measurement-based quantum computing (MBQA), a novel approach to quantum computing which uses projective measurements to achieve universal computation instad of using unitary evolution. Through this project, we introduce the concept of graph states, then formulate an abstract machine on top of this construct. We then provide a prood of universality, and the demonstrate MBQA through a toy problem: Deutsch algorithm.

Method

The main structure used by measurement-based quantum compuitng is a *graph state*[1]. Graph states are multi qubit systems which can be described by a simple graph. For a graph $G = \{V, E\}$, its graph state is defined by the following set of commutative, Hermitian operators it is stabilized by.

$$K = \left\{ K_a = \sigma_x^a \prod_{b \in N_a} \sigma_z^b \,\middle|\, a \in V \right\}, \tag{1}$$

where N_a is the neighborhood of vertex a. Such a graph state can be generated via the following procedure.

1. For each vertex in V, initialize the corresponding qubit with the positive σ_x eigenstate $|+\rangle$.
2. For each edge $(a, b) \in E$, apply the conditonal phase flip gate $S_{ab} = |0\rangle\langle 0| \otimes \mathbb{1} + |1\rangle\langle 1| \otimes \sigma_z$ to the system.

An abstract machine, one-way quantum computer[5], can be constructed with a *cluster*, a graph state with a regular lattice structure that acts a substrate to perform computations, and a *random access measurement device* to measure the qubits in the cluster. The computer takes a measurement pattern

$$\mathcal{M} = \{\vec{r}_a \mid a \in V\}, \tag{2}$$

composed of a measurement direction for each qubit, as its input program.

Circuit-based computers use deterministic unitary evolution for program evolution while one-way computers use probabilistic projective measurements. Even then, these two have the same computational power. This fact can be directly proven by providing a mechanism to simulate unitary evolution using the one-way computer.

One-way computer can simulate a unitary U up to some Pauli element called the *by-product operator* U_Σ[5] that is parameterized by previous measurement outcomes.

$$U' = U_\Sigma U \quad \text{is simulated.} \tag{3}$$

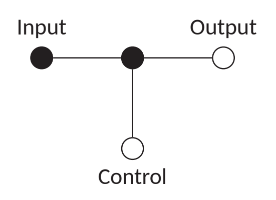
Note that since measurement outcomes are classical data, byproduct operators can be handled by the computer's controller via adapting the next measurement basis during program execution. Any unitary unitary circuit can be rewritten using the universal gates CNOT and arbitrary single-qubit rotation $R(\alpha, \beta, \gamma)$[2]. These two can be implemented by the following operations. $R(\alpha, \beta, \gamma) = R_x(\alpha) R_y(\beta) R_z(\gamma)$ is an arbitrary Euler rotation. Let

$$\mathcal{B}(\varphi) = \left\{ \frac{|0\rangle + e^{i\varphi}|1\rangle}{\sqrt{2}}, \frac{|0\rangle - e^{i\varphi}|1\rangle}{\sqrt{2}} \right\} \tag{4}$$

be a parameterized projection basis. Measuring a 4-qubit linear cluster in the basis $\mathcal{B}(\alpha)$, $\mathcal{B}((-1)^{s_1}\beta)$ and $\mathcal{B}((-1)^{s_2}\gamma)$ implements

$$\underbrace{\sigma_x^{s_2+s_4} \sigma_z^{s_1+s_3}}_{U_\Sigma} R(\alpha, \beta, \gamma). \tag{5}$$

For CNOT, we use the graph state below which implementes $\sigma_{x,\text{out}}^{s_2} \sigma_{z,\text{out}}^{s_1} \sigma_{z,\text{ctrl}}^{s_2}\, \mathrm{CNOT}$ where filled nodes are measured on σ_x basis.



Lastly, we introduce two useful identities. Firstly, measuring a qubit in σ_z basis effectively removes it from the cluster with a by-product σ_x^s. Secondly, measuring a qubit in σ_x in a wire teleports its state to the next vertex.

Results

This project had two phases. The first phase involves the theoretical investigation of measurement-based quantum computing (MBQC). During this phase we reviewed graph states, stabilizer formalism. Furthermore, we introduced the one-way quantum computer as an abstract machine and given a constructive proof of the universality of MBQC, whcih is a fundamental property.

Next, we survayed various simulators to test this framework as there are no real quantum computing backends utilizing MBQC that are available to public use. At the time of writing, there are two alternative simulators. Open sourced `quantumcore`[4] library, which was reported to be used for simulating Deutsch-Jozsa algorithm using MBQC, lacks a public driver and lack any kind of documentation so we were not able to utilize it. The other alternative is a new addition to `pebble quantum`[3], a quantum machine learning platform. Even though we were not able to successfully test a multi-bit function types via Deutsch-Jozsa algorithm, we were able to test single-bit functions using the simpler Deutsch algorithm. Oracle implementations and simulation outcomes are given in Table 1.

Function	Implementation	Type	Outcome
$f(x) = 0$	$\mathbb{1}$	Constant	0
$f(x) = 1$	$\sigma_x^{(0)}$	Constant	0
$f(x) = x$	CX	Balanced	1
$f(x) = \neg x$	$\sigma_x^{(0)} CX$	Balanced	1

Table 1: Function Implementations Used for Deutsch Oracles

References

[1] Hein M. et al. "Entanglement in graph states and its applications". In: *Proceedings of the International School of Physics "Enrico Fermi"* 162.Quantum Computers, Algorithms and Chaos (2006), pp. 115–218. ISSN: 0074-784X. DOI: 10.3254/978-1-61499-018-5-115. arXiv: quant-ph/0602096.

[2] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2009. DOI: 10.1017/cbo9780511976667.

[3] *Paddle Quantum*. 2020. URL: https://github.com/PaddlePaddle/Quantum.

[4] *quantumcore*. https://github.com/fwcd/quantumcore. [Online; accessed 2-Jan-2022]. 2018.

[5] Robert Raussendorf, Daniel E. Browne, and Hans J. Briegel. "Measurement-based quantum computation on cluster states". In: *Phys. Rev. A* 68 (2 Aug. 2003), p. 022312. DOI: 10.1103/PhysRevA.68.022312. arXiv: quant-ph/0301052.