Optimal implementation of quantum unstructured search on existing hardware

Jan Gwinner

BEIT

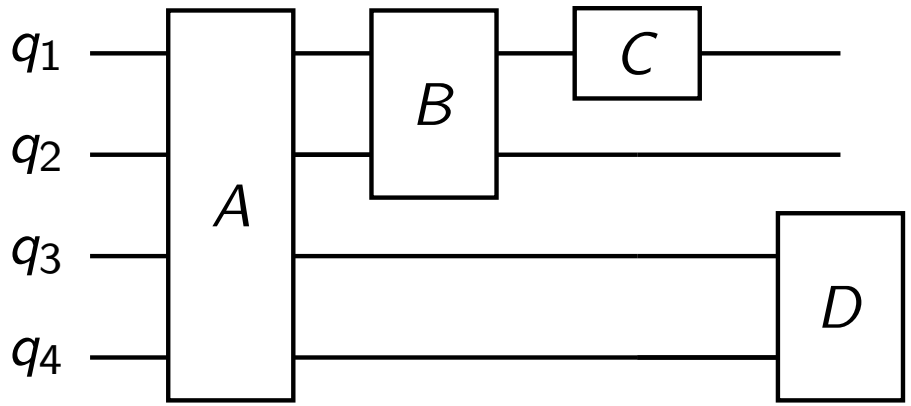
October 12, 2020

Overview

1. Preliminaries
2. Additional gates
3. Partial Uncompute
4. Experiments
5. Bibliography

How to read a quantum circuit?

Assume A, B, C, D to be unitary



This is equivalent to the following operator on $(\mathbb{C}^2)^{\otimes 4}$:

$$(Id_4 \otimes D)(C \otimes Id_8)(B \otimes Id_4)A$$

q_1, q_2, q_3, q_4 are called qubits, A, B, C, D are called gates.

How to program a quantum computer?

To implement a unitary operator U on a quantum computer one first must find its decomposition into a quantum circuit consisting of a restricted set of gates that are supported by a given computer. Currently most native gate sets consist of arbitrary 1-qubit gates and some restricted family of 2-qubit gates.

Connectivity

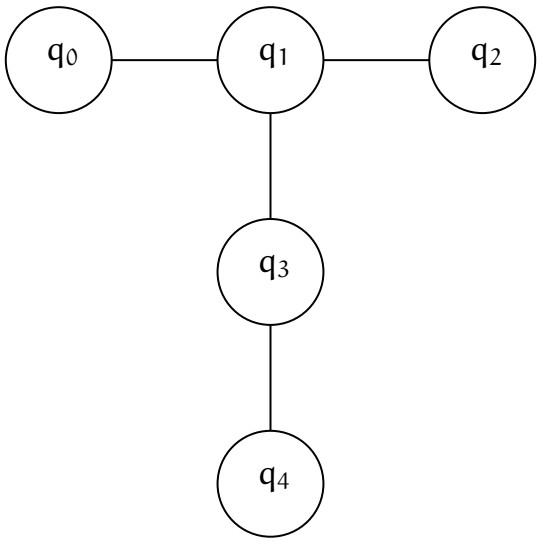
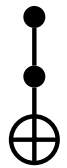


Figure: Topology of IBM Q Vigo quantum processor; edges denote pairs of qubits on which 2-qubit CX gate can be applied

Toffoli gate



denotes Toffoli gate and represents the unitary operator that satisfies for elements of canonical basis:

$$\mathrm{U}|110\rangle = |111\rangle$$
$$\mathrm{U}|111\rangle = |110\rangle$$
$$\mathrm{U}|x\rangle = |x\rangle \text{ , } \textit{otherwise}$$

Quantum Oracle

Consider subset S words over $0, 1$ alphabet of length n. A quantum oracle marking the elements from S is a unitary operator defined on canonical basis of $\mathbb{C}^{2^n}$:

$$O|x\rangle = \begin{cases} -|x\rangle, & \text{if } x \in S \\ |x\rangle, & \text{otherwise} \end{cases}$$

For example if $n = 2$ and $S = \{00, 11\}$ then $O = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$.

For now we will be mostly interested in oracles marking exactly one element.

Unstructured Search Problem

Given some oracle O we want to deduce (by applying a quantum circuit) what elements it marks at the lowest 'cost' possible. We have few possibilities:

- Number of oracles O in quantum circuit
- Number of additional gates
- Number of additional gates + gates required to implement oracles O

Classical solution

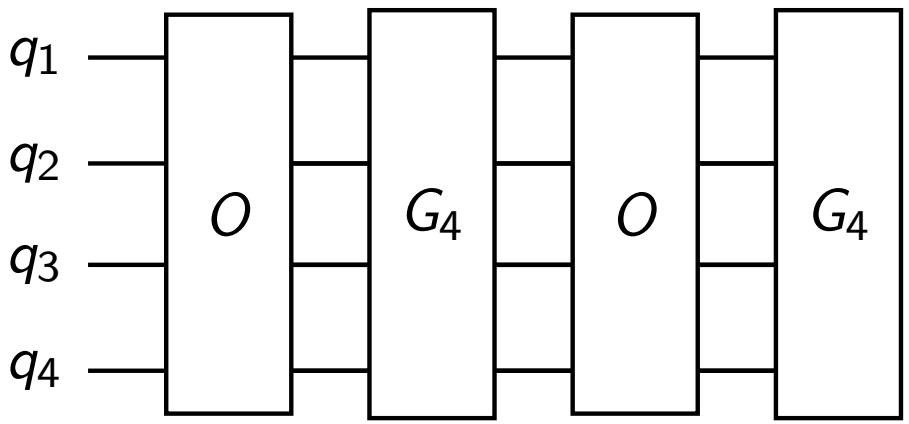
Randomly order the possible words and in this order check if the word is marked by O. Expected number of Oracle calls is therefore 2^{n-1}.

Grover Algorithm pt. 1

The original text can be found here [4]. By G_n we will denote operator $2\left|s\right\rangle\left\langle s\right| - Id$, where $\left|s\right\rangle = \frac{1}{\sqrt{2^n}}\sum_i \left|i\right\rangle$. Assume that the state of machine is $\left|0\ldots0\right\rangle$. We begin by applying Hadamard gate, i.e. $\frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}$ on each qubit to obtain equal superposition $\left|s\right\rangle$. After that we apply the operator OG_n approximately $\frac{\pi}{4}2^{n/2}$ times to achieve probability of measuring the marked element close to 1. This number of oracles in quantum circuit is minimal possible as proved by [9]. Number of non-oracle gates is $\Theta(n2^n)$.

Grover Algorithm pt. 2

For $n = 4$



Amplitude Amplification

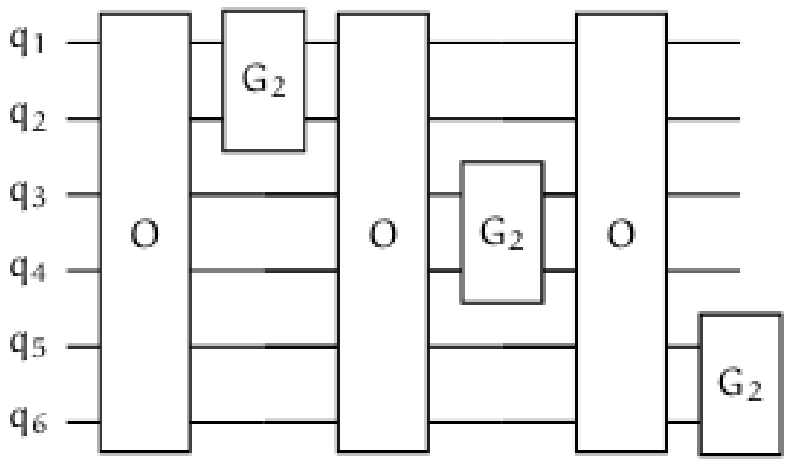
Theorem ([2], p.7 Theorem 2)

Let $\mathcal{A}$ be any quantum algorithm operating on n qubits that uses no measurements, and let $f : \{0,1\}^n \to \{0,1\}$ be any boolean function with a corresponding phase oracle O. Let a be the probability that measuring $\mathcal{A}|00\ldots0\rangle$ yields $|t\rangle$ such that $f(t) = 1$, and assume that $a \in (0,1)$. Let $\theta \in (0, \pi/2)$ be such that $(\sin\theta)^2 = a$, and let $s = \lfloor \frac{\pi}{4\theta} \rfloor$. Then measuring $(-\mathcal{A}F_0\mathcal{A}^\dagger O)^s \mathcal{A}|00\ldots0\rangle$ yields $|t\rangle$ such that $f(t) = 1$ with probability at least $\max\{1-a, a\}$, where

$$F_0|t\rangle = \begin{cases} |t\rangle, & \text{if } t \neq 0 \\ -|t\rangle, & \text{if } t = 0. \end{cases}$$

Previous results pt 1

In [5] Grover introduced the construction that allows to reduce the number of additional gates (i.e. non-oracle gates). The main idea is to use smaller G_n operators.



By replacing 2 with $\log(n)$ and inserting the given superposition into Amplitude Amplification procedure we obtain the marked element using $\Theta(\log n 2^{n/2})$ additional gates.

Previous results pt 2

In 2015 Arunachalam and De Wolf improved this result to $\Theta(\log \log^* n 2^{n/2})$ though the algorithm is complicated and allows actual improvement only for $n \geq 25$. [1] At the end of the paper they ask if there is a way to reduce the number of additional gates to $\Theta(2^{n/2})$ and if there is a way to asymptotically reduce the number of additional gates for oracles marking more than one element. We will answer positively for both questions.

W_j circuits

In [3] we introduce the family of circuits W_j for a sequence $k = (k_1, k_2, k_3, \ldots, k_m)$

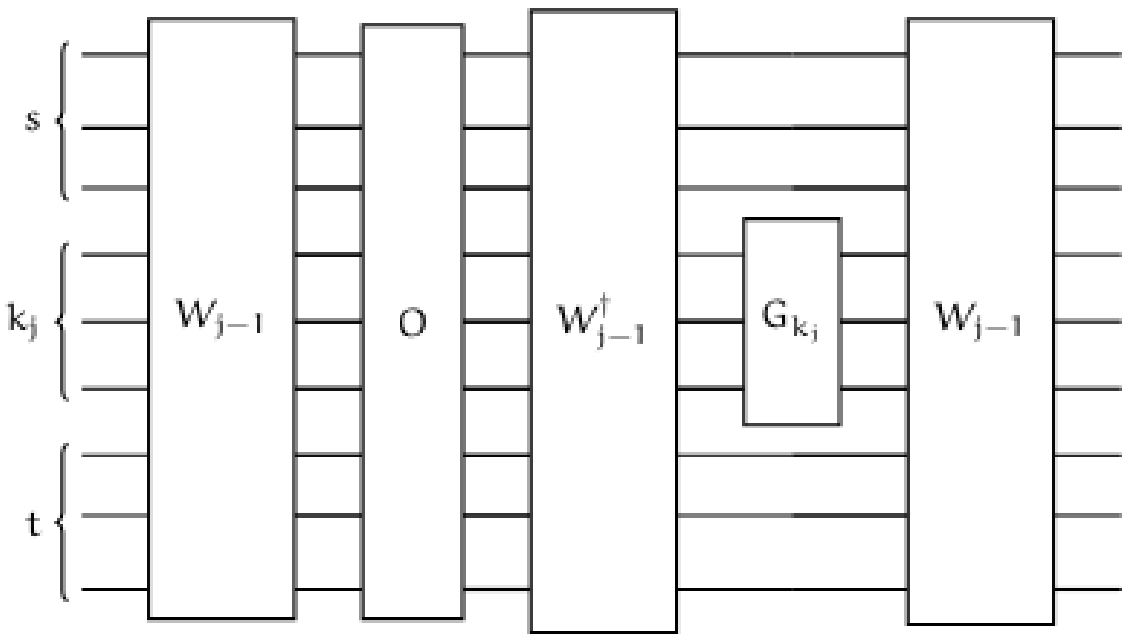


Figure: Graphical representation of the W_j circuit. Note that the oracle in W_{j-1} manipulates all of the qubits, however no other gate does so. In this picture $s = k_1 + \cdots + k_{j-1}$ and $t = k_{j+1} + \cdots + k_m$.

Example W_2

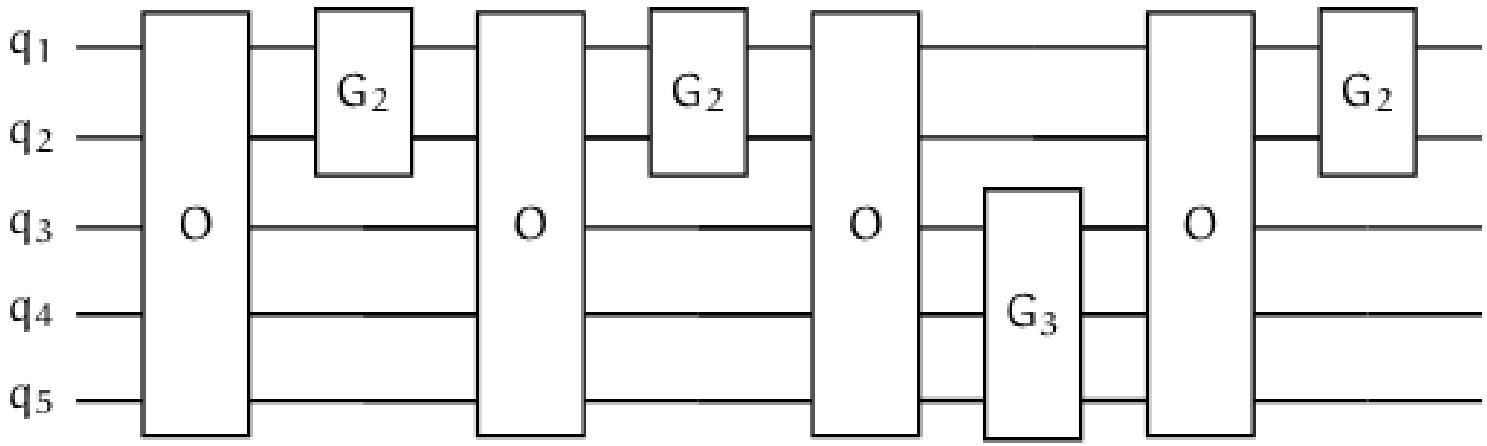


Figure: W_2 for $k = (2, 3)$

Main algorithm

After applying circuit W_k and using Amplitude Amplification we obtain that for $k = (1, 2, 3, \ldots)$ we can find the marked element using $\Theta(2^{n/2})$ additional gates. There are more families of circuits that scale with the same number of additional gates, though the main question is if we may explore this further?

Quantum oracle revisited

Simple visual example of the basic oracle that marks a single element

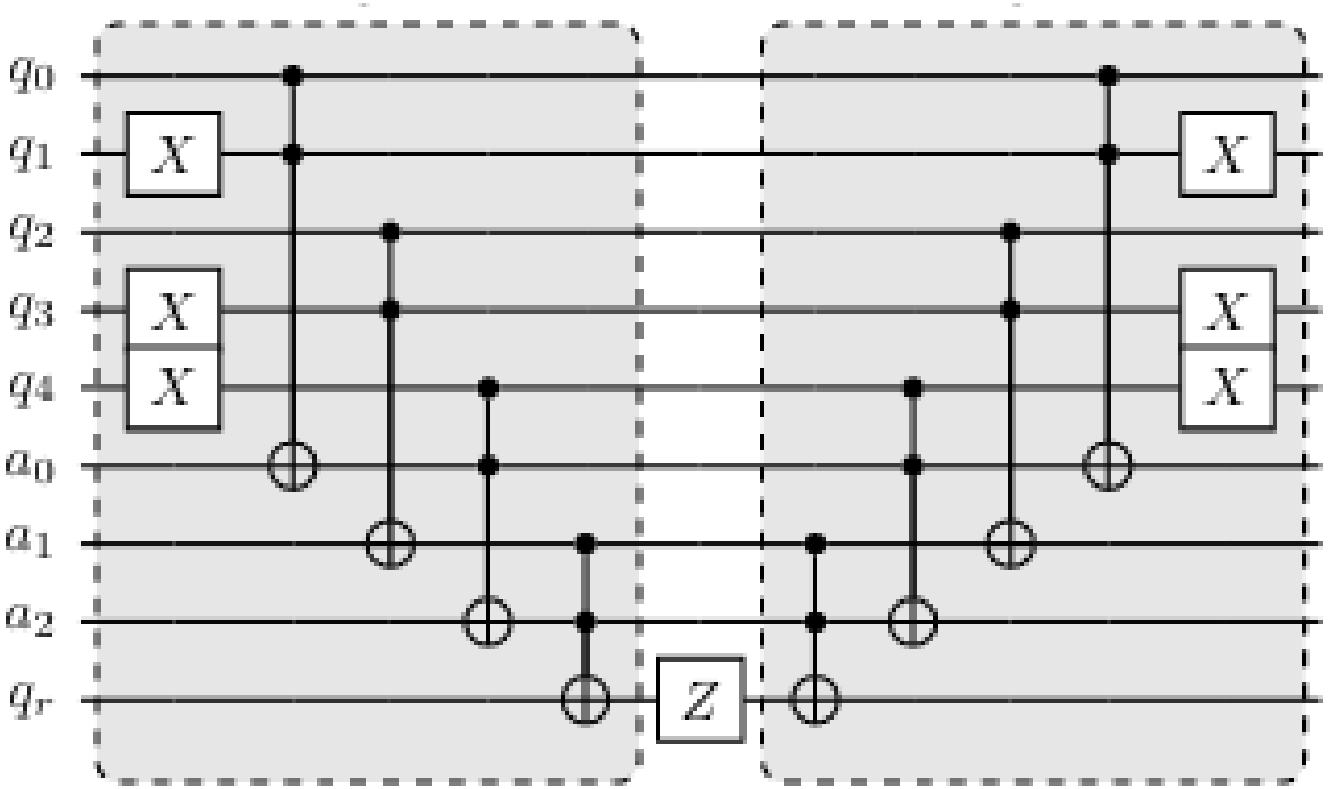


Figure: Simple implementation of quantum oracle marking element $|10100\rangle$ - assuming that a_0, a_1, a_2, q_r are initially in state $|0\rangle$

Intertwined G_2

Operator G_2 that acts on qubits q_2, q_3

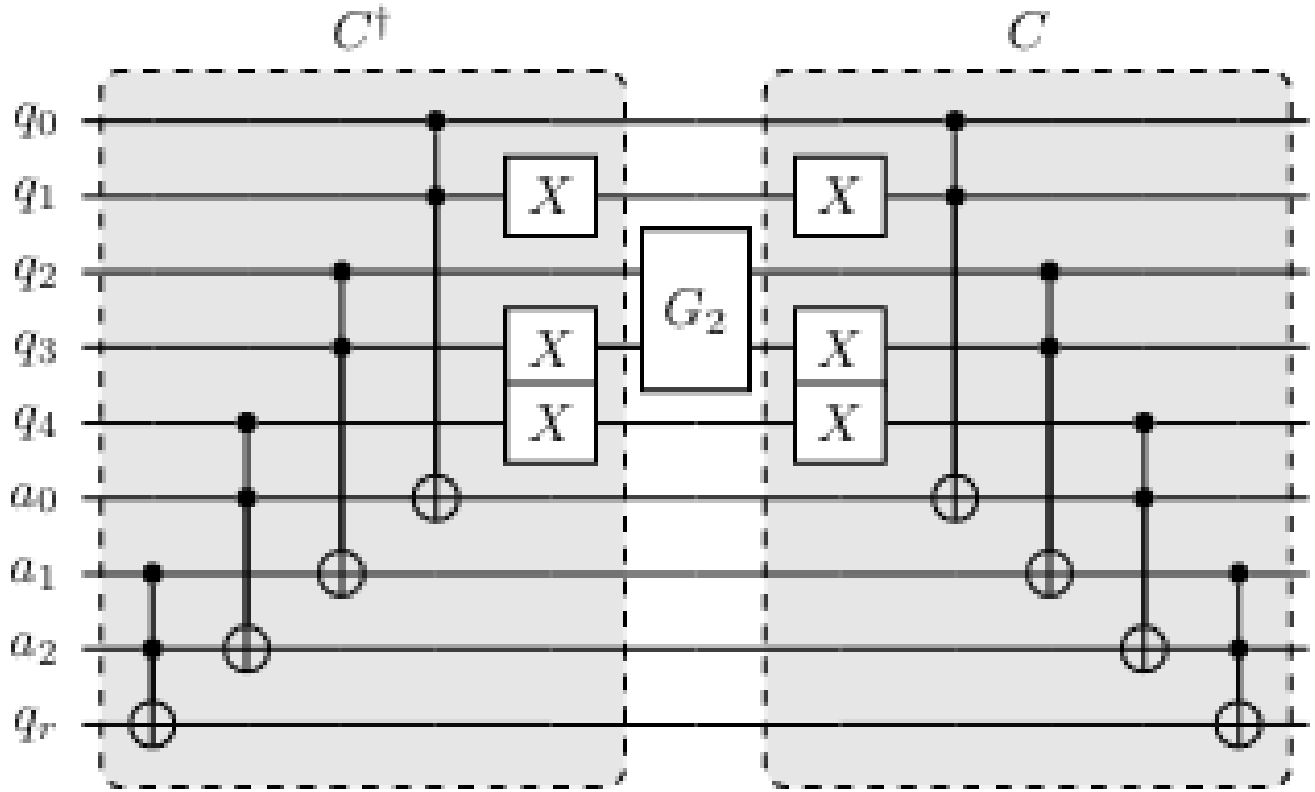


Figure: G_2 between two oracles

Partial uncompute

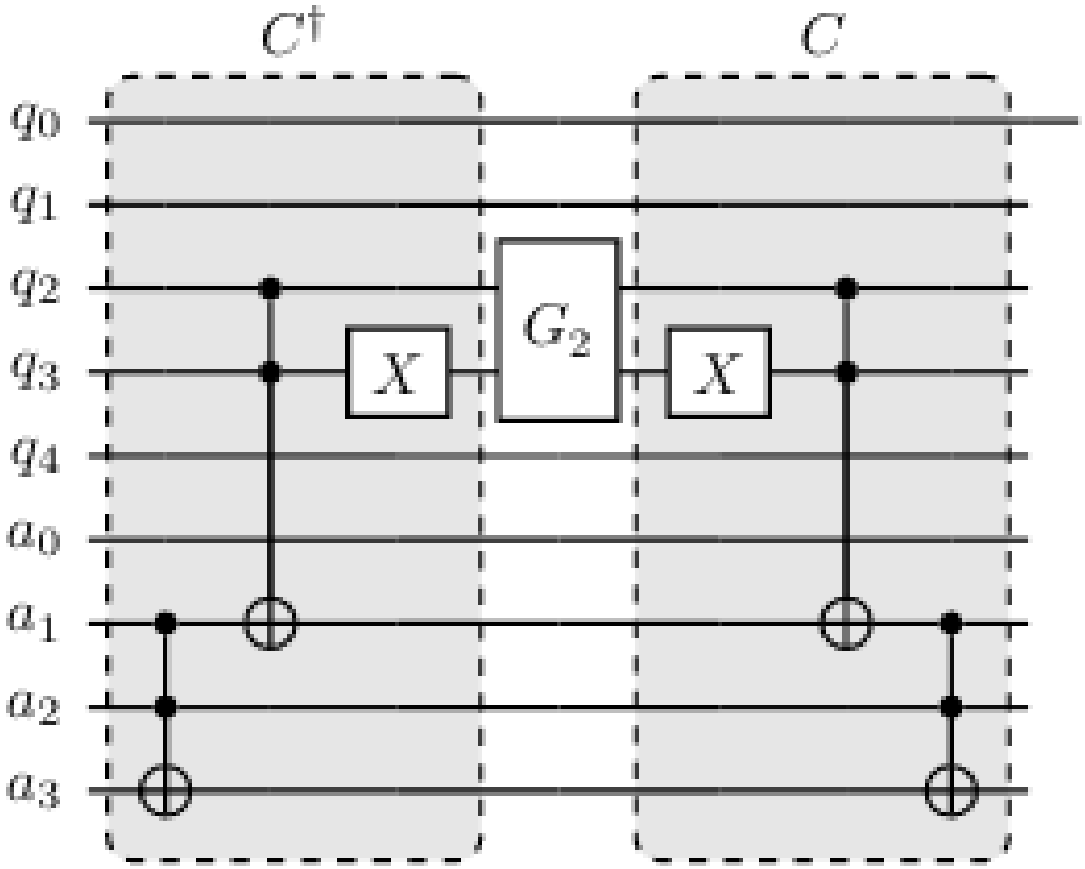


Figure: G_2 between two oracles

Is it useful?

Unstructured search can be applied to any NP problem - for example for Unique k-SAT there exists a quantum circuit that uses $O(m \log(m) 2^{n/2}/n)$ total (oracle and non–oracle) gates and solves the problem with certainty. Though there exist specialized quantum algorithms that can solve Unique k-SAT in significantly better complexity.

Is it useful in theory?

In Theorem 6 in [3] we provide an algorithm to solve unstructured search problem with K marked elements using $O(\log K \frac{2^{n/2}}{\sqrt{K}})$ which is better than previously best known Amplitude Amplification algorithm that used $O(n\frac{2^{n/2}}{\sqrt{K}})$.

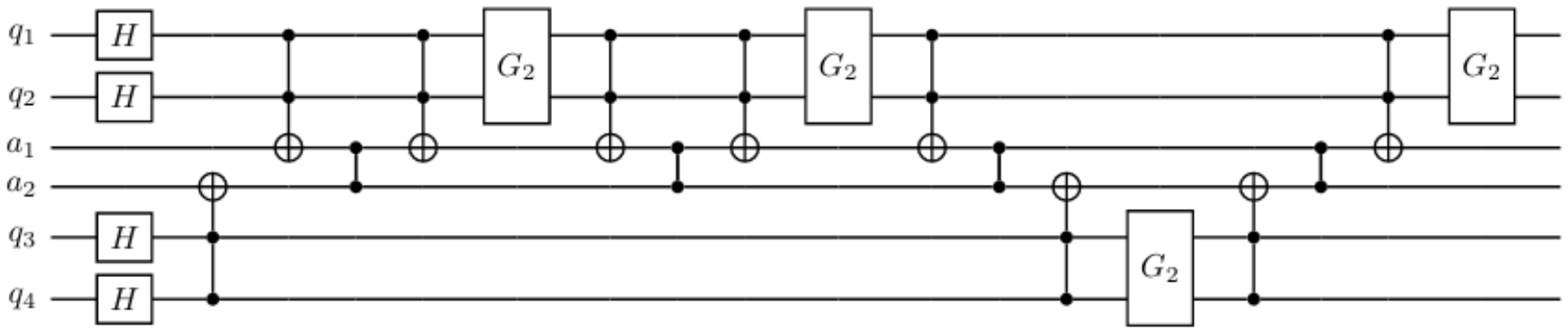
Honeywell System Model H0

The native gate set of Honeywell System Model H0 consists of arbitrary single qubit gates and $U_{zz} = \exp(-i\frac{\pi}{4} Z \otimes Z)$ two qubit gate. The error rate of U_{zz} dominates the error rate of single qubit gates [8]. Note that U_{zz} can be transformed to standard CX gate using only sinle qubit operations, therefore for any circuit that uses k gates CX can be freely transformed into one that uses k gates U_{zz}.

Simplified multi Toffoli

Standard implementation of Toffoli gate on 3 qubits requires 6 CX gates. Though if we are only interested in marking ancilla qubit then we can implement it using only 3 CX gates. This gate is often called a Margolus gate. Standard implementation of Toffoli gate on 4 qubits requires 14 CX gates. Though if we are only interested in marking ancilla qubit then we can implement it using only 6 CX gates. Therefore wherever possible we use simplified implementation.

W_2 circuit on 4 qubits optimized for H0



Results

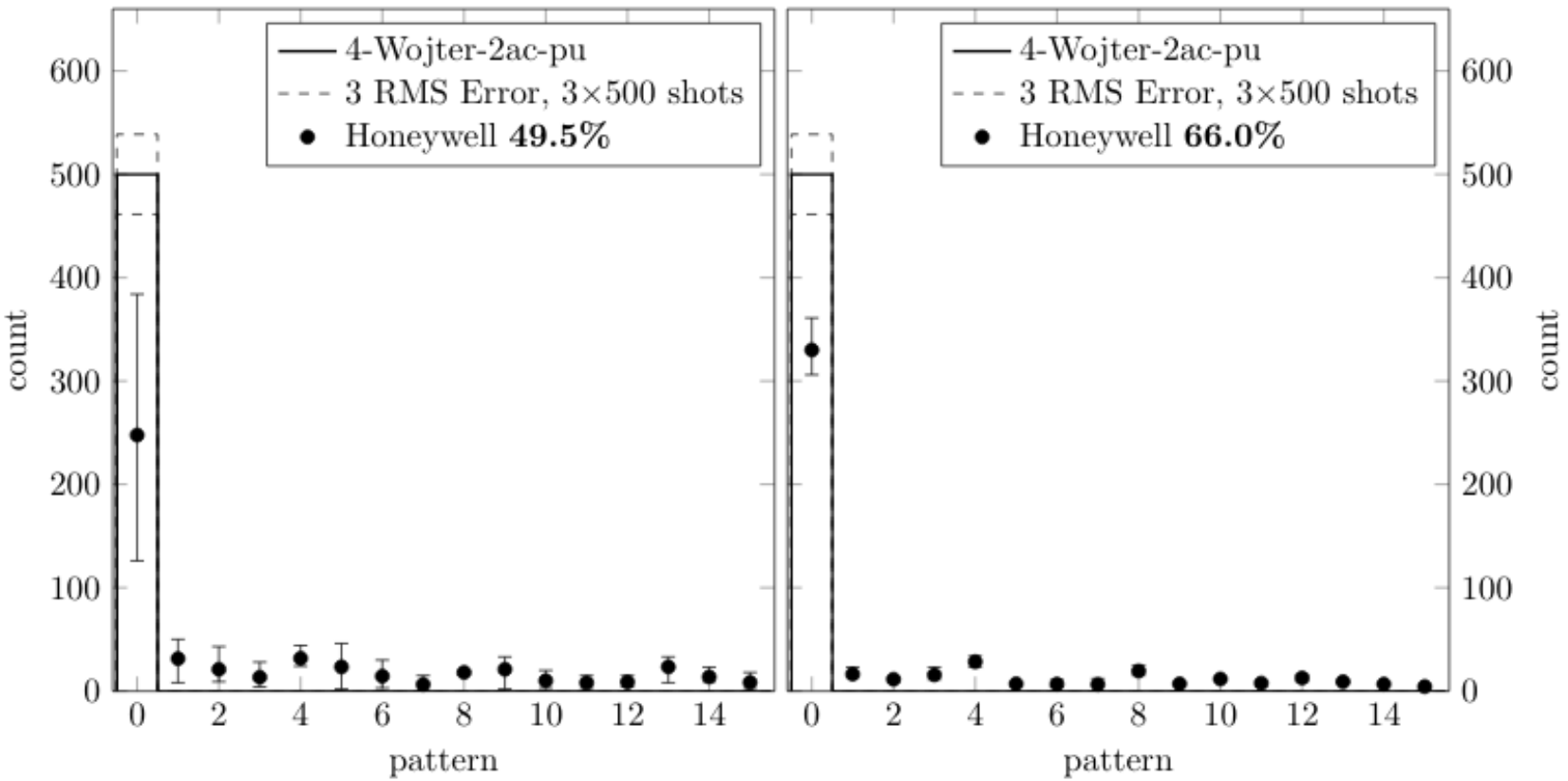
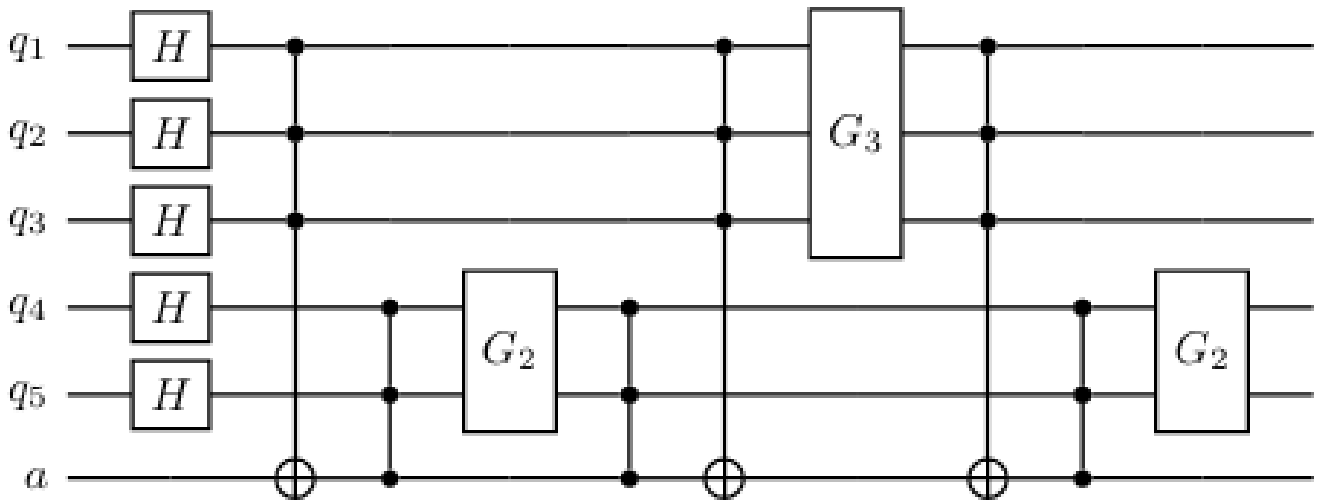


Figure: First and second session results of W_2 on 4 qubits from [7]

D_2 circuit on 5 qubits optimized for H0



Results

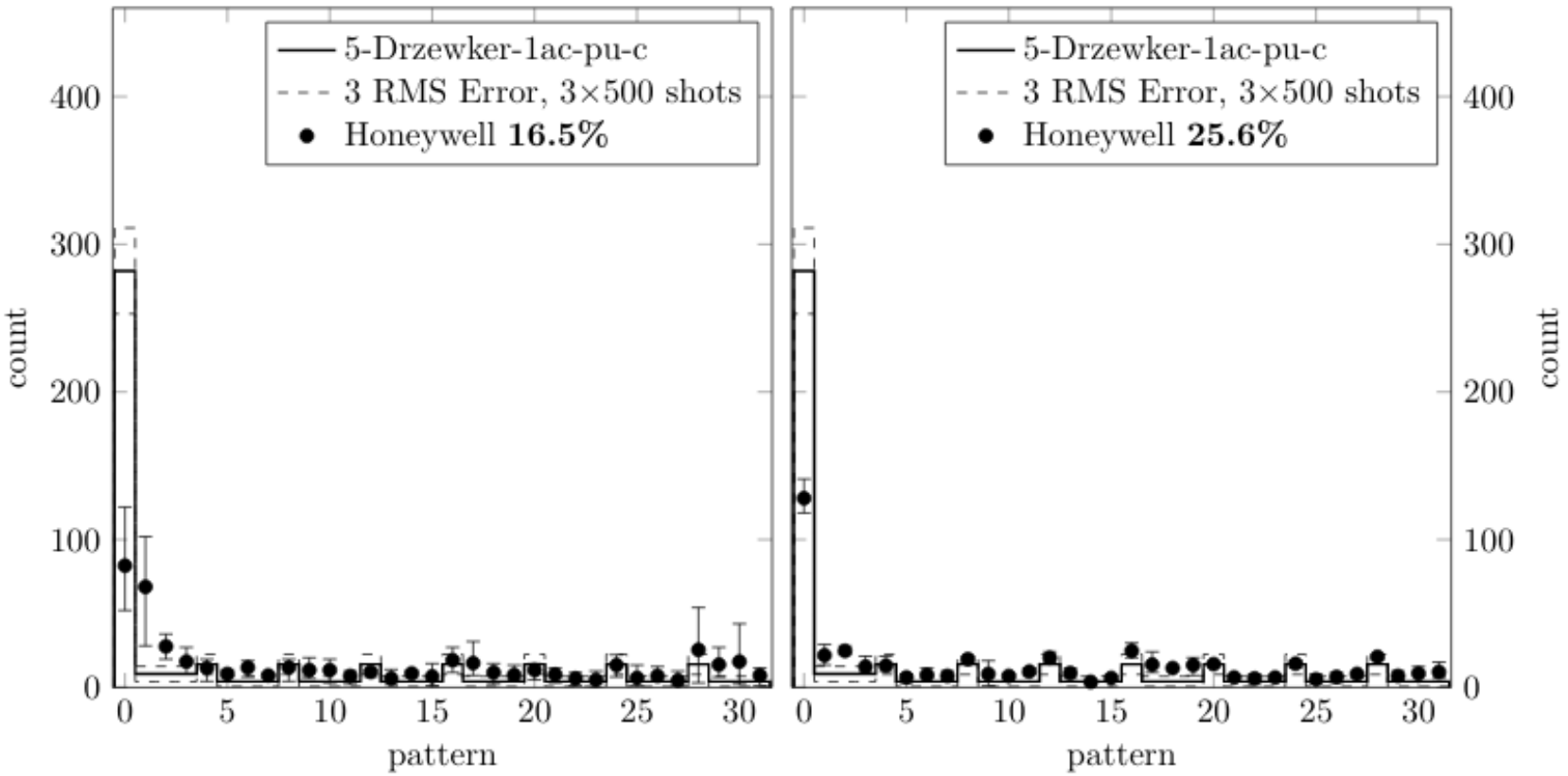
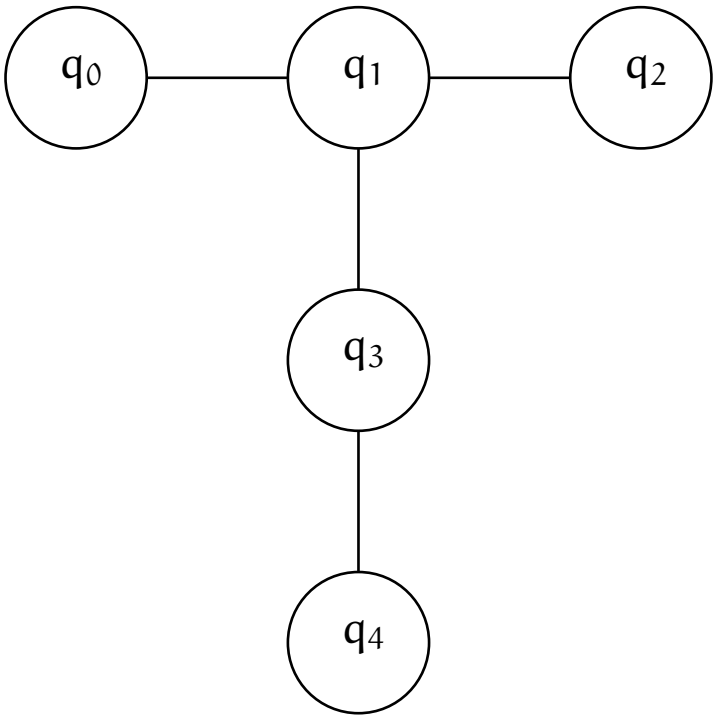


Figure: First and second session results of D_2 on 5 qubits from [7]

IBMQ Vigo

The native set of gates on IBMQ Vigo Quantum Processor consists of all single qubit gates and CX gate on edges of the following graph:



Margolus

Topology aware implementation of Margolus gate:

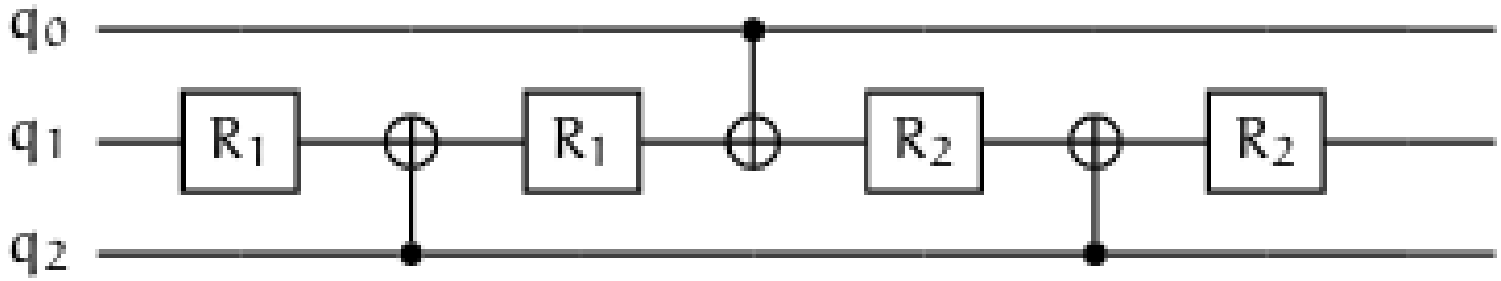


Figure: Margolus gate, where $R_1 = U_3(\frac{\pi}{4}, 0, 0)$ and $R_2 = U_3(-\frac{\pi}{4}, 0, 0)$

Toffoli pt. 1

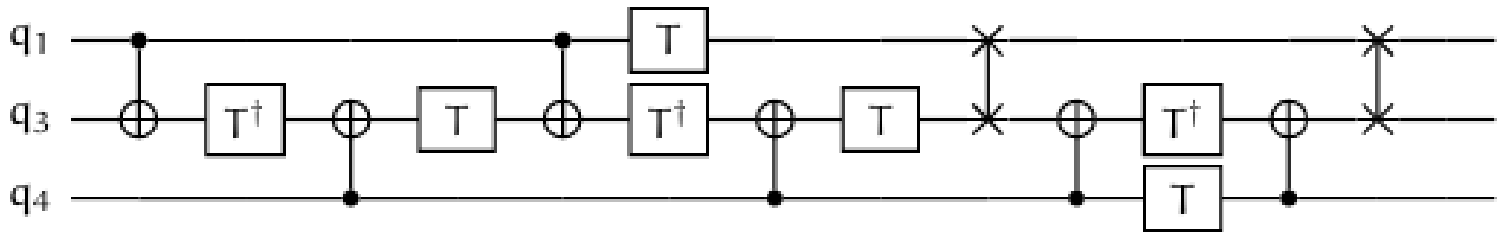


Figure: Naïve implementation of CCZ on qubits q_1, q_3, q_4 in line topology

Toffoli pt. 2

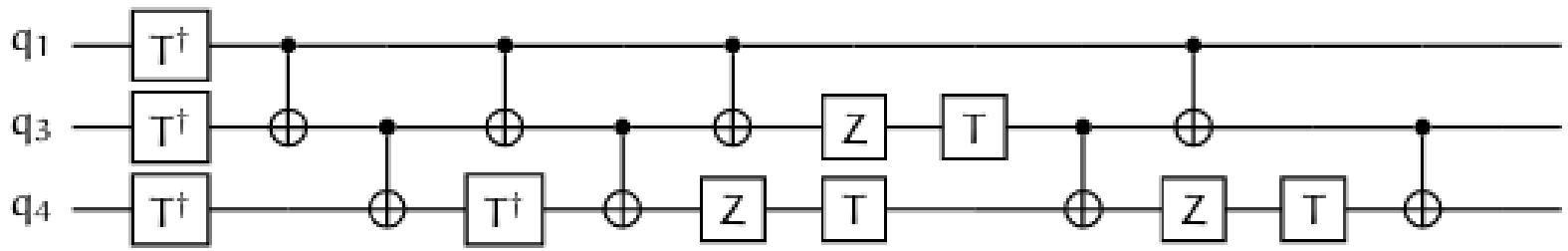


Figure: Improved implementation of CCZ on qubits q_1, q_3, q_4 in line topology

Multi Toffoli

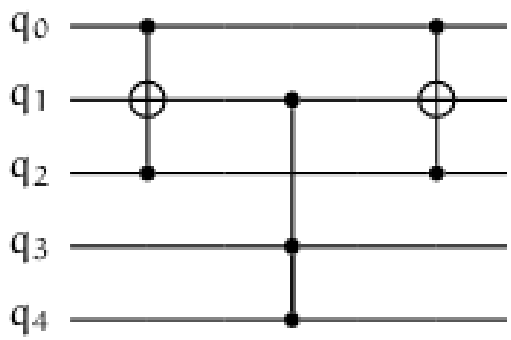


Figure: Improved implementation of $CCCZ$ on qubits q_0, q_2, q_3, q_4 on IBMQ Vigo

Results

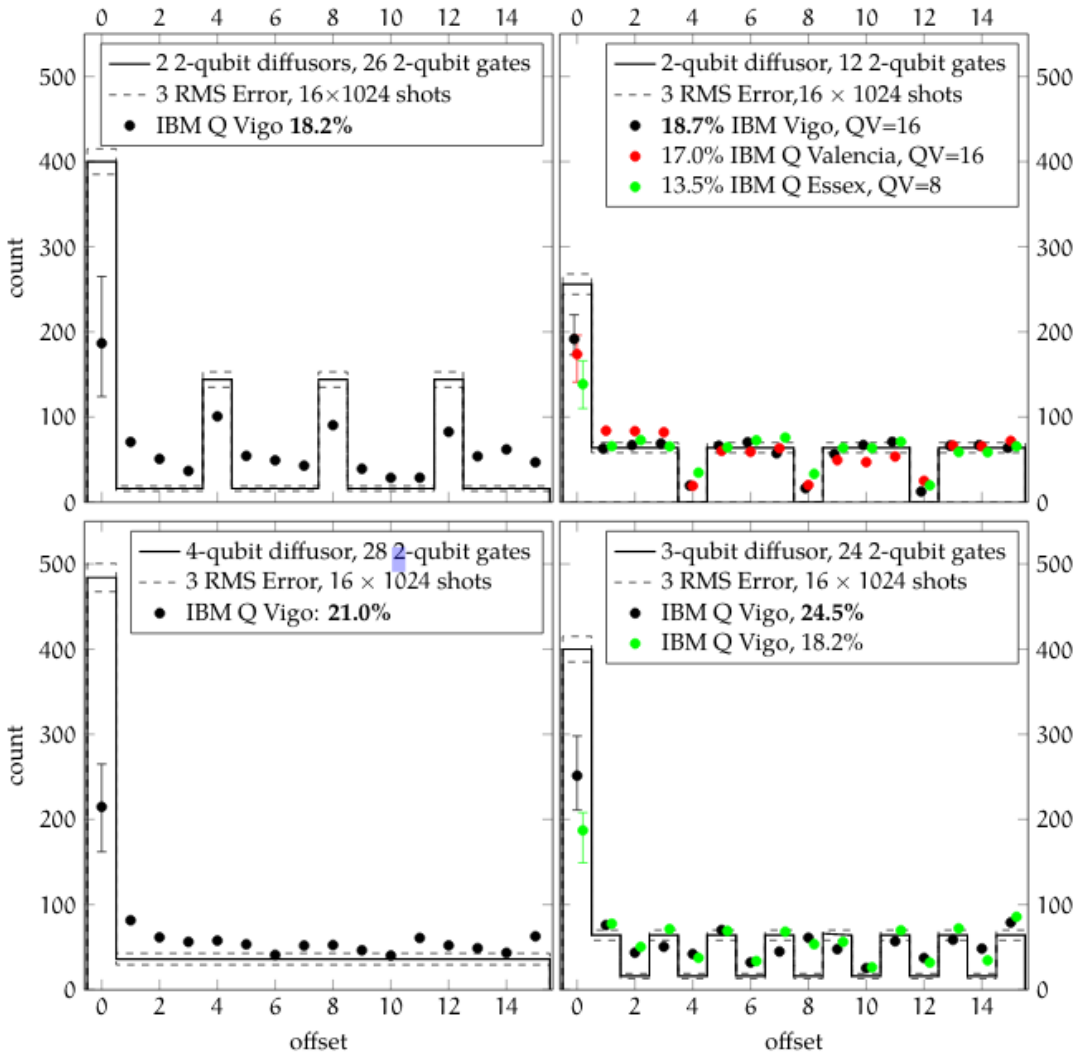


Figure: Results of various searches from [6]

References I


Srinivasan Arunachalam and Ronald De Wolf. "Optimizing the number of gates in quantum search". In: (2015).


Gilles Brassard et al. "Quantum amplitude amplification and estimation". In: (2002).


Marcin Briański et al. "Introducing Structure to Expedite Quantum Search". In: (2020).


Lov K Grover. "A fast quantum mechanical algorithm for database search". In: 1996.


Lov K Grover. "Trade-offs in the quantum search algorithm". In: (2002).


Jan Gwinner et al. "Benchmarking 16-element quantum search algorithms on IBM quantum processors". In: (2020).

References II


Vladyslav Hlembotskyi et al. "Efficient unstructured search implementation on current ion-trap quantum processors". In: (2020).


J. Pino et al. "Demonstration of the QCCD trapped-ion quantum computer architecture". In: (2020).


Christof Zalka. "Grover's quantum searching algorithm is optimal". In: (1999).