REDUCING COMPUTATIONS IN QUANTUM WALK ALGORITHMS

ASIF SHAKEEL

Abstract. Quantum walks (QWs) are of interest as examples of uniquely quantum behavior and are applicable in a variety of quantum search and simulation models. Implementing QWs on quantum devices is useful from both points of view. We describe a prototype one-dimensional QW algorithm that economizes resources required in its implementation. Our algorithm needs only a single shift (increment) operation. It also allows complete flexibility in choosing the shift circuit, a resource intensive part of QW implementations. This is desirable for Noisy Intermediate-Scale Quantum (NISQ) devices, in which fewer computations implies faster execution and reduced effects of noise and decoherence. We implement versions of the algorithm, with two different shift circuit structures, on publicly accessible IBM quantum computers.

Quantum walks (QWs) are studied for exhibiting walk properties that are fundamentally quantum mechanical and for their applications such as in quantum simulation models [1, 2] and quantum search algorithms [3]. They have been experimentally realized [4, 5]. The reader is referred to [6] for a comprehensive review. With public availability of quantum computers, quantum algorithms can be tested and their resource requirements and performance examined. In the current era of Noisy Intermediate-Scale Quantum (NISQ) [7] devices, sources of degradation in the performance of quantum algorithms on a quantum computer include decoherence and noise, both of which can severely limit any gains to be had from a quantum algorithm. As a consequence, the issue of resource use in implementation deserves consideration. In this paper we give a computational description of a basic one-dimensional QW, that conserves resources and is flexible and scalable. It also illustrates that partial optimizations of an algorithm can be done mathematically prior to circuit implementation on a quantum device.

A QW is specified by a particle walking and scattering on a lattice. We will, for this paper, consider a one dimensional walk on a finite lattice of size \( N \). The state of the QW is specified by its position and velocity. Its position is an element of an \( N \)-dimensional position Hilbert space with basis elements labeled \( |x\rangle \), \( x \in \{0, \ldots, N - 1\} \), and a two dimensional velocity Hilbert space with basis elements labeled \( |v\rangle \), \( v = +1, -1 \), for left and right moving particle respectively. \(^1\) The QW Hilbert space \( \mathcal{H} \) is:

\[
\mathcal{H} = \mathbb{C}^2 \otimes \mathbb{C}^N
\]

with basis elements labeled \( |v\rangle \otimes |x\rangle \). The state of a QW is an element \( \psi \in \mathcal{H} \) of unit norm \( \|\psi\| = 1 \). A QW evolves through two consecutive unitary actions on its state.

\(^1\) We informally say right and left moving to mean increment and decrement of the position coordinate, with the understanding that directions are not meaningful on a finite lattice under \( \mod N \) addition.

E-mail address: asif.shakeel@gmail.com
(i) Scattering operation that acts on the velocity space, i.e., is of the form \( S \otimes I_N \), where \( S \) is a unitary matrix on \( \mathbb{C}^2 \), typically \( [8] \),

\[
S = \begin{bmatrix}
i e^{i\alpha} \sin \theta & e^{i\alpha} \cos \theta \\
e^{i\alpha} \cos \theta & i e^{i\alpha} \sin \theta
\end{bmatrix},
\]

and \( I \) is the \( N \times N \) identity matrix. \(^2\)

(ii) Propagation operation that moves the particle in the direction of the velocity,

\[
\sigma : |v\rangle \otimes |x\rangle \mapsto |v\rangle \otimes |x + v\rangle,
\]

where the addition is modulo \( N \).

Denoting \( \hat{S} = S \otimes I_N \), a step \( T \) of QW evolution is thus:

\[
T = \sigma \hat{S}.
\]

This is the model of QW that we are going to be concerned with.

The rest of this paper is organized as follows. In Section 1 we describe our QW algorithm, proposing a method that replaces the right/left shifts (increment/decrement) needed in the propagation \( \sigma \) by simply a right shift (increment), and implements the control of walk direction using qubit-wise CNOT gates on position qubits controlled by the velocity. We also discuss implementing the shift in this scheme by two techniques, one based on generalized CNOT gates and the other on the Quantum Fourier Transform. In Section 2 we run our QW algorithm on some of the IBM quantum computers for a few lattice sizes and number of steps of evolution, and examine their performances for both the shift methods. Section 3 is the conclusion.

1. An algorithm for QW

We begin with a further assumption that the position state on the lattice is encoded by \( n \) qubits, so \( N = 2^n \). We can write the position label \( x \) in terms of qubit labels \( x_j, j \in \{0, \ldots, n - 1\} \) as

\[
x = \sum_{j=0}^{n-1} 2^j x_j,
\]

for \( x_j \in \{0, 1\} \). This identifies each position label with an \( n \)-bit string. Under this identification, we write a basis element of the position space as \( |x\rangle = |x_{n-1} \ldots x_0\rangle \), with a slight notational abuse. Using the natural ordering induced on the position space basis by values

\(^2\)We use the same symbol \( I \) for all the identity operators, the dimensions being implicit.
of $x$ given by Eq. (1), we can write $|x\rangle$ as a coordinate vector

$$|x\rangle = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ (x) \\ 0 \\ \vdots \end{bmatrix}$$

(2)

where the coordinate indices are displayed in parentheses. We also change the label on the velocity state form $|v\rangle \in \{ |+1\rangle, |−1\rangle \}$ in the QW description above, to encode it by a qubit, identifying $|+1\rangle$ with $|0\rangle$, and $|−1\rangle$ with $|1\rangle$ states of a qubit. Under this identification, we write $|v\rangle \in \{ |0\rangle, |1\rangle \}$, again abusing notation slightly. So the joint velocity-position state is given by $n + 1$ qubits, and a basis element is $|v, x\rangle = |v, x_{n-1} \ldots x_0\rangle$ where all the labels take values in $\{0, 1\}$.

We can now use coordinate vectors of length $2N = 2^{n+1}$, with the top $N = 2^n$ elements for positions corresponding to $v = 0$ and lower $N$ for $v = 1$. So, $|v, x\rangle = |v, x_{n-1} \ldots x_0\rangle$ has the coordinate vector

$$|v, x\rangle = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ (2^n v + x) \\ 0 \\ \vdots \end{bmatrix}$$

with coordinate indices in parentheses on the right and $x$ given by Eq. (1). A right shift on the position space alone is denoted by $X$ (generalizing the symbol for one qubit $X$ gate), and given by the $N \times N$ matrix

$$X = \begin{bmatrix} 0 & 0 & 0 & \cdots & 1 \\ 1 & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & 0 & 0 \\ \vdots & \ddots & 1 & 0 & 0 \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix}$$

(3)

We may check that $X |x\rangle = |x + 1\rangle$, where $|x\rangle$ is as in Eq. (2). The matrix for propagation $\sigma$ on the joint velocity-position space is a controlled shift of position $|x\rangle$ by velocity $|v\rangle$, with
|v⟩ = |0⟩ affecting the right shift and |v⟩ = |1⟩ affecting the left shift. Its matrix is

\[ \sigma = \begin{bmatrix} X & 0 \\ 0 & X^T \end{bmatrix}. \] (4)

Figure 1 shows the circuit for this model of QW with \( \hat{S} \) followed by \( \sigma \).

![Quantum Circuit](image)

Figure 1. A quantum circuit for QW. The control |v⟩ selects X or X^T.

1.1. Implementing the propagation \( \sigma \). The propagation \( \sigma \) in Eq. (4), which is the right (left) shift X (X^T) controlled by |v⟩, has a decomposition that we can implement using a single right shift X in Eq. (3).

Let us first state the definition and a property of Toeplitz matrices, which we apply toward our decomposition. Recall that a Toeplitz matrix is a square matrix with each descending diagonal from left to right a constant, i.e., a \( d \times d \) matrix \( A \) is Toeplitz if

\[
A = \begin{bmatrix}
a_0 & a_{-1} & a_{-2} & \cdots & a_{-(d-1)} \\
1 & a_0 & a_{-1} & \cdots & \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
a_{d-1} & \cdots & a_2 & a_1 & a_0
\end{bmatrix}
\]

for some set of numbers \( \{a_{-(d-1)}, \ldots, a_0, \ldots, a_{d-1}\} \). So if we number the rows and columns of \( A \) by \( \{0, \ldots, d-1\} \), then \( A_{i,j} = a_{i-j} \) for all \( 0 \leq i, j \leq d-1 \). Also recall the definition of an exchange matrix (see Section 1.2.11 of [9]). An exchange matrix is a square matrix with all its entries 0 except those on the anti-diagonal which are all 1. Denote the exchange matrix by \( J \), \(^3\)

\[
J = \begin{bmatrix} 0 & \cdots & 0 & 0 & 1 \\ 0 & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots \\ 0 & \cdots & \cdots & 0 & \vdots \\ 1 & 0 & 0 & \cdots & 0 \end{bmatrix}
\]

The transpose of a Toeplitz matrix can be obtained as follows (Section 4.7 of [9]).

\[
A^T = JAJ.
\]

\(^3\)We use the same symbol \( J \) for all the exchange matrices, and either state the dimensions explicitly or let them be implicit by context.
As the shift matrix $X$ in Eq. (3) is Toeplitz, by the above property, we can express $\sigma$ in Eq. (4) as

$$\sigma = \begin{bmatrix} X & 0 \\ 0 & JX \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & J \end{bmatrix} \begin{bmatrix} X & 0 \\ 0 & X \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & J \end{bmatrix},$$

where $J$ is the $N \times N$ exchange matrix. Let us denote the first (and the last) matrix on the right side product decomposition by $C_v(J)$,

$$C_v(J) = \begin{bmatrix} I & 0 \\ 0 & J \end{bmatrix}.$$

Then $\sigma$ in Eq. (5) may be written

$$\sigma = C_v(J) \circ (I \otimes X) \circ C_v(J).$$

We observe that the exchange matrix $J$ is simply the qubit-wise application of $X$ to each position qubit $|x_i\rangle$ (flipping each $|x_i\rangle$: $|0\rangle \equiv |1\rangle$),

$$J = \bigotimes_{i=1}^n X. \tag{7}$$

Let us denote by $C_v^i(X)$ the $|v\rangle$-controlled $X$ on position qubit $|x_i\rangle$, i.e., a CNOT gate controlled by $|v\rangle$ with target $|x_i\rangle$. By the expression for $J$ in Eq. (7), and because $C_v^i(X)$ and $C_v^j(X)$ commute for all $i, j$, we can write $C_v(J)$ as

$$C_v(J) = \prod_{i=1}^n C_v^i(X). \tag{6}$$

In other words, $\sigma$ in Eq. (6) is the qubit-wise application of a CNOT gate controlled by $|v\rangle$ to each $|x_i\rangle$, $i \in \{0, \ldots, n-1\}$, applied both before and after the shift $X$ has been applied to the position space. This $\sigma$ implementation is in Figure 2.

![Figure 2. n-qubit $\sigma$ implementation using a single shift $X$](image)

Note that the order in which the CNOT gates are applied on either side of $X$ in the circuit of Figure 2 does not matter: they have independent targets and could even be applied in parallel. We have simply chosen one arrangement for the convenience of creating a figure.
By the decomposition just presented, we have substantially decreased resources needed for implementation of \( \sigma \). This is because we use the same shift \( X \) for increment and decrement, instead of explicitly implementing both a controlled increment and decrement circuit. Also, the left/right direction control is affected by \( C^v(J) \), while the shift \( X \) only works as a right shift (increment). This allows us the flexibility of choosing any shift circuit for \( X \). We look at two different implementations of shift \( X \) next.

1.2. Generalized CNOT based implementation of right shift \( X \). The right shift \( X \) can be implemented [10] using generalized CNOT gates as shown in Figure 3.

\[
|x_1\rangle \quad \cdots \quad \cdots \quad X \quad \cdots \quad \cdots \quad |x_n\rangle
\]

\[
|x_2\rangle \quad \cdots \quad \cdots \quad |x_{n-1}\rangle \quad \cdots \quad \cdots \quad |x_n\rangle
\]

Figure 3. Generalized CNOT based \( n \)-qubit shift

Implementations of generalized CNOT gates are described in [11–13]. For \( n = 3 \) the generalized CNOT gate is also called the CCNOT gate or the Toffoli gate. and For \( n > 3 \), a generalized CNOT gate can be implemented by \( O(n^2) \) 2-qubit operations or, alternatively, by \( O(n) \) 2-qubit operations and \( O(n) \) ancillary qubits. ⁴ As the shift circuit requires \( O(n) \) generalized CNOT gates, it would need \( O(n^3) \) 2-qubit operations in the former, and, \( O(n^2) \) 2-qubit operations and \( O(n) \) ancillary qubits in the latter, since these ancillary qubits can be reused among the generalized CNOT gates.

1.3. Quantum Fourier Transform (QFT) based implementation of right shift \( X \). The right shift \( X \) can also be implemented using the \( N = 2^n \) dimensional quantum Fourier transform (QFT) [11], denoted by \( \mathcal{F} \). ⁵

\[
\mathcal{F} : |x\rangle \rightarrow \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i x k / N} |k\rangle,
\]

as

\[
X = \mathcal{F}^{-1} \Omega \mathcal{F},
\]

⁴The bounds of gate counts for generalized CNOT gates and the shift given here are to the author’s knowledge.

⁵This description of the QFT based shift algorithm is by David Meyer.
where $\Omega$ is a diagonal phase multiplication matrix which simplifies to a product of $n$ single qubit phase rotations,

$$\Omega = \begin{bmatrix}
1 & 0 & 0 & \cdots & 0 \\
0 & \omega & \cdots & \cdots & \cdots \\
0 & 0 & \omega^2 & 0 & 0 \\
\vdots & \ddots & \ddots & \ddots & \ddots \\
0 & 0 & \cdots & 0 & \omega^{N-1}
\end{bmatrix} = \begin{bmatrix}
1 & 0 \\
0 & \omega
\end{bmatrix} \otimes \begin{bmatrix}
1 & 0 \\
0 & \omega^2
\end{bmatrix} \otimes \cdots \otimes \begin{bmatrix}
1 & 0 \\
0 & \omega^{N/2}
\end{bmatrix},$$

and $\omega = e^{2\pi i/N}$.

The circuit for QFT is in Figure 4.

\[ \text{Figure 4. Quantum Fourier Transform circuit.} \]

where

\[ R_k = \begin{bmatrix}
1 & 0 \\
0 & e^{2\pi i/2^k}
\end{bmatrix}. \]

Note that, strictly, we need to insert $\lfloor n/2 \rfloor$ swaps (each in turn requiring 3 CNOT gates) at the end of the QFT circuit to get the output ordered correctly. Since we use QFT and the inverse quantum Fourier transform (IQFT) together to achieve the shift $X$ in each step of the walk, there is no need to include the swaps. Indeed, the swaps can be absorbed in $\Omega$, by reversing the order of phase rotation operations on qubits. The advantage of QFT based implementation of shift is that the circuit scales in a uniform manner without needing any ancilla qubits.

Let us estimate the gates needed for this implementation of $X$. QFT [11] and its inverse each require $O(n^2)$ 2-qubit operations and $\Omega$ requires $n$ phase rotations. So the shift requires $O(n^2)$ operations.
2. Simulating the QW Algorithm on IBM Quantum Computers

We test the QW algorithm just described, for each of the shift schemes based on the generalized CNOT gates and QFT, on three IBM quantum computers [14]: IBMQX2, IBM-MQX_London and IBM_16_Melbourne, which are 5, 5 and 14 qubit machines respectively. The reader may find the machine gate and coupling maps in Appendix A Section A.1. IBM Qiskit [15] provides the API to access the computers and develop the the python based code, transpile it, and run it.

The lattice sizes we simulate are 4 and 8 corresponding to $n = 2, 3$ qubits. We simulate the walk for either 1 or 2 steps. For all of the simulations, we use the scattering matrix

$$S = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & i \\ i & 1 \end{bmatrix}.$$ 

After initializing the walk joint velocity-position state, we run the algorithm for the given number of evolutionary steps. Then we measure the final state.

For each experiment, i.e., a set of lattice size, initial state, and steps of evolution, we execute the algorithm 1024 times (called “shots”) to generate a distribution on measured states. The circuit is transpiled (compiled to quantum gates) using optimization level 3 of the transpiler prior to executing each experimental run. At this level, the transpiler takes into account the noise properties and the connectivity of the device.

We repeat these runs, for each experiment and each machine, numerous times. We show the result from the run with the smallest $\ell^1$ distance from the ideal distribution. Note that the $\ell^1$ distance between two probability distributions $P$ and $Q$ over a finite, discrete variable $i \in I$ is

$$\ell^1(P, Q) = \frac{1}{2} \sum_{i \in I} |P(i) - Q(i)|.$$ 

It takes values in $[0, 1]$ range. For each experiment, we group together the plots of the distributions for all the machines, and also the ideal distribution that the QW state would have starting from the same state and after the same number of evolutionary steps. A table after each group of plots records the circuit size, the circuit depth, and the $\ell^1$ distance between the ideal distribution and the ones shown. We point out that the circuits that give the distributions with the smallest $\ell^1$ distance from the ideal do not always have the smallest transpiled size. We record the smallest circuit sizes in Appendix A Section A.4.

Simulations using generalized CNOT based shift. We first test the algorithm with the generalized CNOT based shift. Note that for the cases we consider, with lattice sizes of 4 and 8 ($n = 2, 3$ qubits), the generalized CNOT based shifts use the standard CNOT and CCNOT (Toffoli) gates with well-known and optimized implementations included in the gate sets of IBM Qiskit.

The first set of plots are for $N = 4$ sites ($n = 2$), and 1 step of walk evolution. Recall that the joint velocity-position state is $|v, x\rangle$. For instance, the state $|110\rangle$ has $|v\rangle = |1\rangle$, and $|x\rangle = |10\rangle$. The initial walk states for this set of experiments were chosen to be $|v, x\rangle = |010\rangle$. 

---

6For the terminology in this paper related to the IBM quantum computers, the reader might wish to look up IBM quantum computing website [14].

7The circuit size is the smallest number of overall gate count. The circuit depth is the longest path length in gates from an input to the output.

8The general case in Section 1.2 is for $n > 3$. 

The ideal plot shows that the walk has moved both right and the left by 1 lattice point as it should, as the scattering creates both a left and a right moving component. The measured distributions for IBMQX2 and IBMQX_London have peaks at the correct states, with other states showing up as well. The distribution from IBM_16_Melbourne deviates more than the others from the ideal distribution. The circuit layouts for each case are in Appendix A Section A.2.

The table below summarizes the circuit size and depth, and the $\ell_1$ distance of the measured distribution from the ideal distribution. It shows that different architectures transpile to different sizes (gate counts) and depths. Comparison with the gate coupling maps in Appendix A confirms the highest connectivity gate map, which is that of IBMQX2, corresponds to the lowest circuit size and depth, and the least deviation from the ideal distribution.

|                | (size, depth) | $\ell_1$ distance from ideal |
|----------------|--------------|-------------------------------|
| ibmqx2         | (9, 6)       | 0.0752                        |
| ibmq_london    | (13, 9)      | 0.1621                        |
| ibmq_16_melbourne | (31, 18)   | 0.375                         |

Table 1. CNOT based shift, 4 sites, 1 step

The next set of plots and table are for $N = 4$ sites ($n = 2$), for 2 steps of walk evolution. Initial state of the walk is $|v, x\rangle = |010\rangle$. The patterns resemble those that occur as in the previous case. The circuit sizes and depths increase and the corresponding deviations from the ideal are larger as well. The distribution from IBM_16_Melbourne in particular suffers from strongest deviations.

We can speculate about the causes and sources of the deviations from the ideal. The circuit size and depth, gate noise, leakage, the actual sequence of operations and the decoherence, and other factors, are likely.
The table below shows, as expected, bigger circuit sizes and depths. It is interesting that the $\ell^1$ distance is lesser for IBMQX_London and IBM_16_Melbourne in this case than were in the case of 1 evolution step above.

|                | (size, depth) | $\ell^1$ distance from ideal |
|----------------|---------------|-----------------------------|
| ibmqx2         | (15, 10)      | 0.0928                      |
| ibmq_london    | (44, 27)      | 0.1162                      |
| ibmq_16_melbourne | (47, 30)   | 0.3018                      |

**Table 2.** CNOT based shift, 4 sites, 2 steps

The following set of plots are for $N = 8$ sites ($n = 3$), and 1 step of walk evolution. Initial state of the walk is $|v, x\rangle = |0010\rangle$. While the distributions for IBMQX2 and IBMQX_London show reasonable agreement with the ideal, the distribution from IBM_16_Melbourne is no longer recognizable as being a result of the quantum walk with discernible peaks at the correct states.
The $\ell^1$ distance from ideal for IBM\_16\_Melbourne in the table below is much bigger than for the other machines. The circuit size for IBM\_16\_Melbourne has grown a lot more compared to the previous case than it has for the other machines.

| Machine              | (size, depth) | $\ell^1$ distance from ideal |
|----------------------|---------------|-------------------------------|
| ibmq\_x2             | (33, 21)      | 0.2871                        |
| ibmq\_london         | (60, 37)      | 0.3818                        |
| ibmq\_16\_melbourne | (81, 45)      | 0.7510                        |

**Table 3.** CNOT based shift, 8 sites, 1 step

**Simulations using QFT based shift.** For this set of experiments, we implement the QW algorithm using the QFT based shift. As a reminder, the joint velocity-position state is $|v, x\rangle$. For example, the state $|011\rangle$ has $|v\rangle = |0\rangle$, and $|x\rangle = |11\rangle$. The trends in distributions are similar to the ones observed in generalized CNOT case. We note that the circuit sizes and depths are in generally larger in the present case, accompanied with higher deviations of the measured distributions from the ideal. We surmise it is because the CNOT implementation for $n = 2, 3$ are small and are part of the intrinsic gate sets at the level of IBM Qiskit, in contrast to the QFT based implementation. The circuit layouts for each case are in Appendix A Section A.3.

The following set of plots and the table are for $N = 4$ sites ($n = 2$), for 1 step of walk evolution. Initial state of the walk is $|v, x\rangle = |010\rangle$. 
The next set of plots and the table are for $N = 4$ sites ($n = 2$), for 2 steps of walk evolution. Initial state of the walk is $|v,x⟩ = |010⟩$. 

**Table 4.** QFT based shift, 4 sites, 1 step

|         | (size, depth) | $\ell^1$ distance from ideal |
|---------|---------------|-----------------------------|
| ibmqx2  | (11, 7)       | 0.0781                      |
| ibmq_london | (26, 18)     | 0.2031                      |
| ibmq_16_melbourne | (33, 21) | 0.3301                      |

**Figure 8.** QFT based shift, 4 sites, 1 step

**Figure 9.** QFT based shift, 4 sites, 2 steps
The $\ell^1$ distances in the table mimic the trend in the last section with generalized CNOT based shift. The values are lower for IBMQX_London and IBM_16_Melbourne than those in the 1 step case.

|         | (size, depth) | $\ell^1$ distance from ideal |
|---------|---------------|-----------------------------|
| ibmq2   | (18, 11)      | 0.1836                      |
| ibmq_london | (55, 35)   | 0.1104                      |
| ibmq_16_melbourne | (57, 36) | 0.2695                      |

**Table 5.** QFT based shift, 4 sites, 2 steps

This last set of plots are for $N = 8$ sites ($n = 3$), for 1 step of walk evolution. Initial state of the walk is $|v, x\rangle = |0010\rangle$.

|         | (size, depth) | $\ell^1$ distance from ideal |
|---------|---------------|-----------------------------|
| ibmq2   | (62, 35)      | 0.5098                      |
| ibmq_london | (59, 38)   | 0.5605                      |
| ibmq_16_melbourne | (79, 50) | 0.8047                      |

**Table 6.** QFT based shift, 8 sites, 1 step

Overall, as apparent in the table below, the circuits are larger, respectively, than those encountered so far. The deviations from the ideal are more pronounced as well. IBM_16_Melbourne distribution bears little similarity to the ideal.
3. Conclusion

In this paper we have developed a QW algorithm that uses fewer resources by simplifying the structure of the controlled shift used to implement the propagation part of a QW. A single shift (increment) circuit suffices as opposed to both an increment and a decrement circuit that would usually be needed. The implementation allows any shift circuit, so that an optimized shift circuit dependent on the particular machine architecture may be substituted. In the NISQ regime where noise and decoherence adversely affect performance, this serves as an advantage.

We examined two particular shift circuits. These are the generalized CNOT gate based shift, and the QFT based shift. The advantage that the QFT based shift has is that the resources are well quantified and its 2 qubit gate count scales as $O(n^2)$ where lattice size is $N = 2^n$. The algorithm for both the shift circuits were run on three IBM quantum computers: IBMQX2, IBMQX_London and IBM_16_Melbourne, with 5, 5 and 14 qubit architectures respectively. We were able to simulate QW over small lattice sizes $N = 4, 8$ ($n = 2, 3$) and for 1 and 2 QW evolution steps, with reasonable results. We found that the higher connectivity architectures, like IBMQX2 and IBMQX_London generally perform better. Compiled circuit sizes were smaller for the CNOT gate based shift than the QFT based shift. That is expected for the small $n$ case we simulated, as the CNOT and Toffoli gate (CCNOT) have standard optimized implementations. For large lattice sizes, the QW with QFT based shift may prove more economical, though the transpiler optimizations and the machine architecture and noise properties have a strong influence in determining the optimal circuit size and depth. We executed each run for 1024 shots, noting that a higher a number of shots would yield measurement distributions that are more accurate with respect to the machine behavior, especially for higher $n$.

Future work may seek to understand how the transpiler optimizations could be directed to improve the QW performance on specific architectures. In a similar vein, higher dimensional QWs, and generally quantum algorithms and simulation models could be adapted by mathematical structure to better meet the resource constraints of evolving quantum computers.

Acknowledgments

The author would like to thank David Meyer for the ideas and discussions that motivated, and are partly represented in, this paper.

References

[1] P. Arrighi, G. Di Molfetta, I. Márquez-Martín, and A. Pérez, Dirac equation as a quantum walk over the honeycomb and triangular lattices, Phys. Rev. A 97 (2018), 062111. doi:10.1103/PhysRevA.97.062111
[2] D. W. Berry and A. M. Childs, Black-box Hamiltonian simulation and unitary implementation, Quantum Info. Comput. 12 (2012), 29–62. rintonpress.com/xxqic12/qic-12-12/0029-0062.epsp.
[3] F. Magniez, A. Nayak, J. Roland, and M. Santha, Search via Quantum Walk, SIAM J. Comput. 40 (2011), 142-164. doi:10.1145/1250790.1250874
[4] A. Schreiber, K. N. Cassemiro, V. Potoček, A. Gábris, P. J. Mosley, E. Andersson, I. Jex, and Ch. Silberhorn, Photons Walking the Line: A Quantum Walk with Adjustable Coin Operations, Physical Review Letters 104 (2010), no. 5, 050502. doi:10.1103/PhysRevLett.104.050502
[5] M. Broome, A. Fedrizzi, B. Lanyon, I. Kassal, A. Aspuru-Guzik, and A. White, Discrete Single-Photon Quantum Walks with Tunable Decoherence, Phys. Rev. Lett. 104 (2010), 153602. doi:10.1103/PhysRevLett.104.153602
[6] S Venegas-Andraca, Quantum walks: a comprehensive review, Quant. Inf. Proc. 11 (2012), 1015–1106. doi.org/10.1007/s11128-012-0432-5

[7] J. Preskill, Quantum Computing in the NISQ era and beyond, Quantum 2 (2018), 79. doi:10.22331/q-2018-08-06-79

[8] D. A. Meyer, From quantum cellular automata to quantum lattice gases, Journal of Statistical Physics 85 (1996), 551–574. doi:10.1007/BF02199356

[9] G. Golub and C. Van Loan, Matrix Computations: 4th Edition, Johns Hopkins Studies in the Mathematical Sciences, Johns Hopkins University Press, Baltimore, MD, 2013.

[10] B. Douglas and J. Wang, Efficient quantum circuit implementation of quantum walks, Phys. Rev. A 79 (2009), 052335. doi:10.1103/PhysRevA.79.052335

[11] M. Nielsen and I. Chuang, Quantum Computation and Quantum Information: 10th Anniversary Edition, Cambridge University Press, New York, 2010. doi:10.1017/CBO9780511976667

[12] A. Barenco, C. Bennett, R. Cleve, D. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter, Elementary gates for quantum computation, Phys. Rev. A 52 (1995), 3457–3467. doi:10.1103/PhysRevA.52.3457

[13] Mehdi and Pedram Saeedi Massoud, Linear-depth quantum circuits for n-qubit Toffoli gates with no ancilla, Phys. Rev. A 87 (2013), 062318. doi:10.1103/PhysRevA.87.062318

[14] IBM Quantum Experience, 2019. url:quantum-computing.ibm.com

[15] Qiskit: An Open-source Framework for Quantum Computing, 2019. url:qiskit.org
Appendix A. Coupling maps, layouts and smallest transpiled circuit sizes

A.1. Coupling maps. These are the IBM quantum computer coupling maps for the machines in this paper. The directional arrows show source-target pair for controlled operations. Arrows in both directions mean both qubits can serve as the control and target.

**Figure 11.** ibmqx2 coupling map

**Figure 12.** ibmq_london coupling map

**Figure 13.** ibmq_16_melbourne coupling map
A.2. **Layouts for generalized CNOT based shift.** Here we show the layout maps for each of the experiments with generalized CNOT based shift. The qubits and couplings in black are involved in the respective circuit.

![Figure 14. CNOT based shift, 4 sites, 1 step.](image1)

![Figure 15. CNOT based shift, 4 sites, 2 steps.](image2)

![Figure 16. CNOT based shift, 8 sites, 1 step.](image3)
A.3. **Layouts for QFT based shift.** Here we show the layout maps for each of the experiments with QFT based shift. The qubits and couplings in black are involved in the respective circuit.

**Figure 17.** QFT based shift, 4 sites, 1 step.

**Figure 18.** QFT based shift, 4 sites, 2 steps.

**Figure 19.** QFT based shift, 8 sites, 1 step.
A.4. Smallest transpiled circuit sizes. The smallest circuit sizes that were obtained during simulations are given below for the respective experiments. These are not always the same as the circuits that gave the least $\ell^1$ distance from the ideal distribution shown in the main body of the paper.

| Circuit          | (size, depth) |
|------------------|---------------|
| ibmqx2           | (9, 6)        |
| ibmq_london      | (13, 9)       |
| ibmq_16_melbourne| (26, 15)      |

Table 7. CNOT based shift, 4 sites, 1 step

| Circuit          | (size, depth) |
|------------------|---------------|
| ibmqx2           | (15, 10)      |
| ibmq_london      | (27, 22)      |
| ibmq_16_melbourne| (41, 26)      |

Table 8. CNOT based shift, 4 sites, 2 steps

| Circuit          | (size, depth) |
|------------------|---------------|
| ibmqx2           | (33, 21)      |
| ibmq_london      | (43, 31)      |
| ibmq_16_melbourne| (58, 38)      |

Table 9. CNOT based shift, 8 sites, 1 step

| Circuit          | (size, depth) |
|------------------|---------------|
| ibmqx2           | (11, 7)       |
| ibmq_london      | (21, 15)      |
| ibmq_16_melbourne| (25, 16)      |

Table 10. QFT based shift, 4 sites, 1 step

| Circuit          | (size, depth) |
|------------------|---------------|
| ibmqx2           | (18, 11)      |
| ibmq_london      | (38, 29)      |
| ibmq_16_melbourne| (48, 30)      |

Table 11. QFT based shift, 4 sites, 2 steps
| Machine                | (size, depth) |
|------------------------|---------------|
| ibmqx2                 | (47, 33)      |
| ibmq_london            | (59, 38)      |
| ibmq_16_melbourne      | (78, 45)      |

**Table 12.** QFT based shift, 8 sites, 1 step