Computation at a Distance

Samuel A. Kutin* David Petrie Moulton* Lawren M. Smithline*

May 4, 2007

Abstract

We consider a model of computation motivated by possible limitations on quantum computers. We have a linear array of n wires, and we may perform operations only on pairs of adjacent wires. Our goal is to build a circuits that perform specified operations spanning all n wires. We show that the natural lower bound of $n-1$ on circuit depth is nearly tight for a variety of problems, and we prove linear upper bounds for additional problems. In particular, using only gates adding a wire (mod 2) into an adjacent wire, we can realize any linear operation in $\mathrm{GL}_n(2)$ as a circuit of depth $5n$. We show that some linear operations require depth at least $2n+1$.

1 Introduction

We consider the following model of computation: We have n wires, labeled $\langle 1\rangle$ through $\langle n\rangle$. Each wire carries a single bit. We are allowed to perform reversible linear operations on adjacent wires: $\langle i\rangle \oplus= \langle i+1\rangle$ or $\langle i\rangle \oplus= \langle i-1\rangle$. We assume throughout that n is at least 2.

Our goal is to perform some calculation spanning all n wires; for example, we might want to set $\langle n\rangle \oplus= \langle 1\rangle$ and leave the other $n-2$ wires unchanged. Our primary measure of complexity is the *depth* of a circuit; we will also consider the *size* of the circuit (that is, the number of gates).

The motivation for this problem is quantum circuit design. In some proposed models of quantum computation [2, 4, 7, 8], we can perform operations only on adjacent bits, so it is important to consider the cost of computing with bits separated by a given distance. Since the eventual topology of quantum computers is unknown, we choose to focus on linear arrays of bits. Results here should at least be applicable to other topologies.

We note that our model is wholly classical; there are no quantum operations. To perform a quantum gate, one could first move bits around using classical operations and then apply the quantum gate to adjacent bits. We discuss the cost of this approach in Section 3.1.

It is often helpful to take an algebraic view of these circuit problems. We adopt the convention that the wires of our circuit contain column vectors, and we describe the state of all of the wires by the matrix whose ith column is the contents of wire $\langle i\rangle$. A CNOT gate adds the vector on one wire into the vector on another wire. Any circuit performs a series of column operations; note that circuits act on the *right*.

Any function on n bits that can be built out of additions may be viewed as an element of the group $\mathrm{GL}_n(2)$ of $n \times n$ invertible matrices over the field $\mathbf{F}_2$ of two elements. A single gate is

*Center for Communications Research, 805 Bunn Drive, Princeton, NJ 08540.
Email: {kutin,moulton,lawren}@idaccr.org

represented by an elementary matrix consisting of the identity matrix with a single 1 either just above or just below the main diagonal. These matrices generate the group, so we can build any reversible linear operation on our wires using these gates.[1]

It is not hard to show that any element of $\mathrm{GL}_n(2)$ can be constructed from $O(n^2)$ gates, that is, as a product of $O(n^2)$ of the above generators. A simple counting argument gives a lower bound of $\Omega(n^2/\log n)$ for generic circuits. In Section 7.4, we give a lower bound of $(1-o(1))n^2$ for generic elements of $\mathrm{GL}_n(2)$.

Our primary complexity measure is depth, rather than size, so the generating set of interest is different. We allow any set of 1s just off the diagonal, as long as all the row and column indices are distinct; we cannot have two gates using the same wire at the same time. All of our questions can be rephrased in this setting: What is the shortest product of these generators equal to a particular element of the group?

We label the wires by $\langle 1\rangle$ through $\langle n\rangle$ and their initial values by a_1 through a_n. In our diagrams, we draw the wires horizontally, with time proceeding from left to right, wire $\langle 1\rangle$ at the top, and wire $\langle n\rangle$ at the bottom. We analyze the costs of the following problems:

Add Perform $\langle n\rangle = a_1 \oplus a_n$; for each other i, leave $\langle i\rangle = a_i$.

Swap Set $\langle n\rangle = a_1$ and $\langle 1\rangle = a_n$; for each other i, leave $\langle i\rangle = a_i$.

Rotate Set $\langle n\rangle = a_1$; for each $i < n$, set $\langle i\rangle = a_{i+1}$.

Reverse Set $\langle n+1-i\rangle = a_i$ for each i.

Permute Set $\langle \sigma(i)\rangle = a_i$ for each i, given some $\sigma \in S_n$.

Compute Apply an arbitrary $M \in \mathrm{GL}_n(2)$ to the n wires.

The first two tasks require us to perform an operation on $\langle 1\rangle$ and $\langle n\rangle$, leaving the other bits untouched. The next three tasks require us to reorder the bits; this might be useful if a quantum circuit will perform complex calculations on different subsets of the bits. The final task encompasses any possible linear computation.

It is immediate that each of these tasks requires depth $n-1$, since we need to move the information in a_1 at least $n-1$ times.[2] We encourage the reader to work out low-depth solutions to the above problems before reading further.

We will prove the following results. In each case, our proof is via an explicit construction.

Theorem 1.1. *We can add across n wires in depth $n+4$.*

Theorem 1.2. *We can swap across n wires in depth $n+8$.*

Our swapping circuit works by moving a_1 and a_n to two adjacent wires in depth roughly $n/2$, swapping the values, and then moving the wires back. Instead of swapping the values, we could apply any 2-qubit gate to the two wires. So, we can apply any 2-qubit quantum gate spanning n wires in depth $n+O(1)$. In Section 3.1, we will generalize the above argument. We can apply any m-qubit gate whose total span is at most n in depth $n+O(m)$.

[1] To implement reversible affine operations, we would need to allow unary negation gates as well. All such negations could be deferred to one final time-slice.

[2] For permutation and arbitrary computation, this lower bound applies in the worst case.

Theorem 1.3. *We can rotate n wires in depth $n + 5$.*

Theorem 1.4. *We can reverse n wires in depth $2n + 2$.*

We will show in Section 5.2 that reversal requires a depth of at least $2n + 1$.

Theorem 1.5. *For any $\sigma \in S_n$, there is a circuit implementing σ of depth at most $3n$.*

Theorem 1.6. *For any $M \in \mathrm{GL}_n(2)$, there is a circuit implementing M of depth at most $5n$.*

We will show in Section 7.4 that, for any $\epsilon > 0$, almost every matrix in $\mathrm{GL}_n(2)$ requires depth at least $(2 - \epsilon)n$. One natural problem is to close the gap between this lower bound and the upper bound of $5n$. We discuss this, and other open questions, in Section 8.

2 Addition

Theorem 2.1. *We can add across n wires in depth $n + 3$ for even n and in depth $n + 4$ for odd n. The circuit has size $4n - 7$.*

An example of the construction for $n = 10$ appears in Figure 1.

Proof. Let $k = \lceil n/2 \rceil$. We will construct a subcircuit of depth $k + 1$ and size $2n - 4$ that has the following effects:

1. $\langle k \rangle = a_1$.
2. a_n contributes only to wire $\langle k + 1 \rangle$.

Next, we perform $\langle k + 1 \rangle \oplus= \langle k \rangle$; this just replaces a_n by $a_n \oplus a_1$ in the only location where a_n appears. Finally, we undo the subcircuit. When we are done, we have $\langle n \rangle = a_n \oplus a_1$, and each other wire has its initial value. The overall circuit size is $4n - 7$, and the depth is $2k + 3$ as desired.

It remains only to discuss the subcircuit, which is described in Figure 2. We begin with the first two loops, or "cascades". The first loop writes $a_i \oplus a_{i+1}$ to $\langle i \rangle$ for $i < k$. After the second loop, $\langle i \rangle$ contains $a_1 \oplus a_{i+1}$ for $i < k$, and $\langle k \rangle$ contains a_1. Notice that we can start the second loop during the third time-slice, so the two cascades together have depth $k + 1$.

The third and fourth loops can be similarly analyzed. After both loops are completed, we have written a_{i-1} to $\langle i \rangle$ (for $i > k + 1$) and $\bigoplus_{j=k+1}^{n} a_j$ to $\langle k + 1 \rangle$. As desired, a_n affects only $\langle k + 1 \rangle$. The depth is $(n - 1 - k) + 2 \leq k + 1$. □

3 Swap

Theorem 3.1. *We can swap across n wires in depth $n + 7$ for even n and in depth $n + 8$ for odd n. The circuit has size $6n - 9$.*

An example of this construction for $n = 9$ appears in Figure 3.

Proof. We use the same basic idea as in the proof of Theorem 2.1. As before, let $k = \lceil n/2 \rceil$. Before, we built a subcircuit guaranteeing that $\langle k \rangle = a_1$ and that a_n contributes only to wire $\langle k + 1 \rangle$. For a swap, we need something stronger:

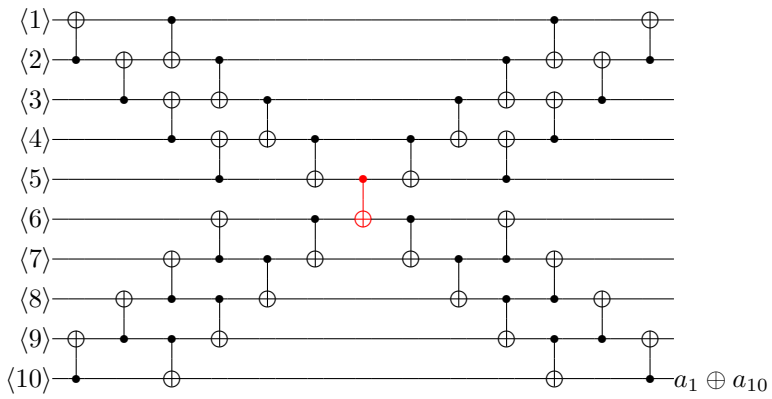

Figure 1: Addition across 10 wires ($k = 5$) in depth 13. The central CNOT is shown in red.

for $i = 1$ **to** $k - 1$:
    $\langle i \rangle \oplus= \langle i + 1 \rangle$
for $i = 1$ **to** $k - 1$:
    $\langle i + 1 \rangle \oplus= \langle i \rangle$
for $i = n - 1$ **down to** $k + 1$:
    $\langle i \rangle \oplus= \langle i + 1 \rangle$
for $i = n - 1$ **down to** $k + 1$:
    $\langle i + 1 \rangle \oplus= \langle i \rangle$

Figure 2: Subcircuit for addition. We take $k = \lceil n/2 \rceil$.

1. $\langle k \rangle = a_1$.
2. $\langle k + 1 \rangle = a_n$.
3. No other wire depends on a_1 or a_n.

Our subcircuit will have size $3n - 6$ and depth $k + 3$.

We begin by running the subcircuit. Next, we swap $\langle k \rangle$ and $\langle k + 1 \rangle$; this requires three gates. Finally, we undo the subcircuit. The overall size is $6n - 9$.

The subcircuit is described in Figure 4. The first two loops are the same as in Figure 2. We write $a_1 + a_{i+1}$ to $\langle i \rangle$ (for $i < k$) and a_1 to $\langle k \rangle$. The next loop erases the a_1 information; when it concludes, we have $\langle i \rangle = a_{i+1} + a_{i+2}$ for $i < k - 1$, $\langle k - 1 \rangle = a_k$, and $\langle k \rangle = a_1$. As before, we can nest the cascades (see Figure 3); the depth is $k + 3$.

The remaining loops are similar. After the penultimate loop, we have $\langle i \rangle = a_{i-1}$ for $i > k + 1$ and $\langle k + 1 \rangle = \bigoplus_{j=k+1}^{n} a_j$. The final loop accumulates upward; we obtain $\langle i \rangle = \bigoplus_{j=i-1}^{n-1} a_j$ for $i > k + 1$, and $\langle k + 1 \rangle = a_n$. The depth is $(n - 1 - k) + 4 \leq k + 3$.

Since the subcircuit has depth $k + 3$, and the central swap has depth 3, one might think the overall depth would be $2k + 9$. In fact, we can reduce the depth to $2k + 7$. Two of the three gates in the swap commute with adjacent gates and can be nested into the subcircuit, as shown in Figure 3. □

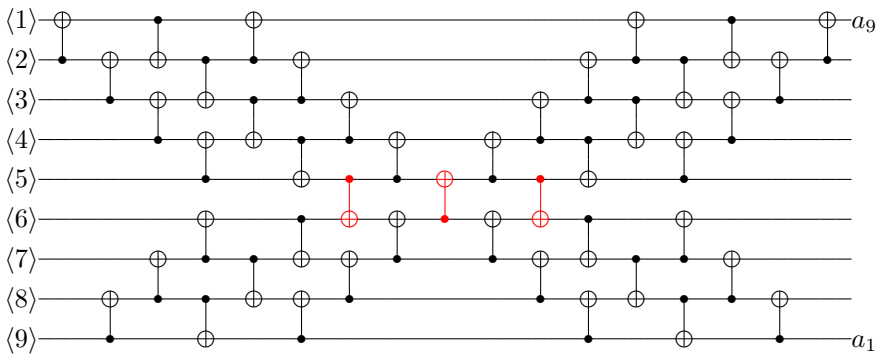

Figure 3: Swap across 9 wires ($k = 5$) in depth 17. The central swap is shown in red.

for $i = 1$ **to** $k - 1$:
    $\langle i \rangle \oplus= \langle i + 1 \rangle$
for $i = 1$ **to** $k - 1$:
    $\langle i + 1 \rangle \oplus= \langle i \rangle$
for $i = 1$ **to** $k - 1$:
    $\langle i \rangle \oplus= \langle i + 1 \rangle$
for $i = n - 1$ **down to** $k + 1$:
    $\langle i \rangle \oplus= \langle i + 1 \rangle$
for $i = n - 1$ **down to** $k + 1$:
    $\langle i + 1 \rangle \oplus= \langle i \rangle$
for $i = n - 1$ **down to** $k + 1$:
    $\langle i \rangle \oplus= \langle i + 1 \rangle$

Figure 4: Subcircuit for swap. We take $k = \lceil n/2 \rceil$.

3.1 Arbitrary Quantum Gates

As noted in the Introduction, we could replace the central swap with any operation on a_1 and a_n; in the quantum setting, we could use any 2-qubit gate. Hence, any 2-qubit gate spanning n wires can be implemented in depth $n + O(1)$.

Suppose that we wish to implement an m-qubit gate with span n. We need to operate on a set of bits $\langle i_1 \rangle, \ldots, \langle i_m \rangle$ with $1 = i_1 < i_2 < \cdots < i_m = n$. Write $b_\ell = a_{i_\ell}$. Let $k = \lceil n/2 \rceil$ as above, and choose j with $i_j \leq k < i_{j+1}$.

For each ℓ between 1 and m, we will move b_ℓ onto the wire $\langle k - j + \ell \rangle$, so the bits will lie on m adjacent wires. We then perform the m-qubit gate. Finally, we undo the transformation.

We will begin with nested cascades as in our swap circuit; we use the top half of the subcircuit of Figure 4, but we only let i range from i_j to $k - 1$. When we finish, we have $\langle k \rangle = b_j$, and no other wire depends on b_j. The wires between $\langle j \rangle$ and $\langle k - 1 \rangle$ contain some complicated functions of various a_i bits, but none of the b_ℓ bits are involved.

Next, if $j > 1$, we perform cascades moving b_{j-1} to $\langle k - 1 \rangle$. We continue, performing a series of j sets of cascades; the final set moves b_1 into $\langle k - j + 1 \rangle$. Since the cascades nest, the total depth is $k + O(m)$.

At the same time, we perform upward cascades moving b_{j+1} to $\langle k + 1 \rangle$, b_{j+2} to $\langle k + 2 \rangle$, and so on, up to moving b_m to $\langle k - j + m \rangle$. After $k + O(m)$ time-slices, we have moved the m bits of interest onto the wires from $\langle k - j + 1 \rangle$ to $\langle k - j + m \rangle$.

Finally, we perform the m-qubit gate, and we reverse the first part of the computation to put all the bits back. The overall depth is $n + O(m)$, in addition to the cost of the m-qubit quantum gate.

Moreover, suppose we wish to perform several long-range gates spanning n wires, and using a total of m bits, simultaneously. We first move those m bits together in depth $n + O(m)$. Next, we permute the bits in depth $O(m)$ (see Section 6), so the bits for each gate are adjacent. We now perform the quantum gates and then undo the rest of the calculation. The total depth is again $n + O(m)$, in addition to the cost of the most complicated quantum gate.

4 Rotation

Recall that rotating n wires means setting $\langle n \rangle$ to a_1 and setting $\langle i \rangle$ to a_{i+1} for each other i.

Theorem 4.1. *For $n > 2$, we can rotate n wires in depth $n + 5$. The circuit has size $4n - 6$.*

We first give a rotation circuit of depth $2n + 1$. We then explain how to use this circuit in our main construction. An example of the final result with $n = 10$ is depicted in Figure 5.

Lemma 4.2. *We can rotate n wires in depth $2n + 1$. The circuit has size $4n - 5$.*

Proof. We consider the rotation circuit of Figure 6, which we call $R(\ell, m)$.

The first three loops of $R(\ell, m)$ are similar to those in Figure 4. After the first loop, we have $\langle j \rangle = \bigoplus_{i=\ell}^{j} a_i$ for $\ell \leq j \leq m$. The second loop leaves $\langle m \rangle = \bigoplus_{i=\ell}^{m} a_i$, and sets each other $\langle j \rangle$ to a_{j+1}. The third loop sets $\langle j \rangle$ to $\bigoplus_{i=\ell+1}^{j+1} a_j$ for $j < m$, but sets $\langle m \rangle = a_\ell$. The final loop restores $\langle j \rangle$ to a_{j+1} for $j < m$.

The circuit $R(\ell, m)$ contains $4(m - \ell) - 1$ gates. The first three loops can be nested, for a combined depth of $(m - \ell) + 4$. The total depth is $2(m - \ell) + 3$. If we take $\ell = 1$ and $m = n$, we obtain a rotation of all n wires. □

Note that if we flip each gate in $R(\ell, m)$ upside-down, the resulting circuit still performs a rotation. More generally, the circuit formed by flipping each gate of a given circuit upside-down performs the inverse transpose of the $\mathrm{GL}_n(2)$ transformation performed by the original circuit.

Proof of Theorem 4.1. Let $k = \lceil n/2 \rceil$. We let $R(\ell, m)$ be the circuit of Lemma 4.2.

We let $R'(\ell, m)$ be the circuit $R(\ell, m)$ run upside-down and backward. Note that running a rotation circuit upside-down makes it rotate in the opposite direction, and running any circuit backward makes it perform the inverse operation. So, $R'(\ell, m)$ has the same effect as $R(\ell, m)$.

We define a circuit C as follows:

1. Apply $R(1, k)$.
2. Apply $R'(k, n)$.

First, note that the first half of C sets $\langle j \rangle = a_{j+1}$ for $j < k$, and $\langle k \rangle = a_1$. Consequently, the second half of C completes the rotation. So, C rotates n wires as desired. Clearly the size of C is $4n - 6$.

The only bit used both by $R(1, k)$ and $R'(k, n)$ is $\langle k \rangle$. Note that $R(1, k)$ is done accessing bit $\langle k \rangle$ after $k + 3$ time-slices, and $R'(k, d)$ does not access $\langle k \rangle$ until time-slice $n - k$. Hence, the total depth of the circuit is $(k + 3) + ((n - k) + 4) = n + 7$.

We can further reduce the depth to $n + 5$. The last access of $\langle k \rangle$ by $R(1, k)$ and the first access by $R'(k, n)$ both write to $\langle k \rangle$. These two operations commute with each other. By swapping the order, we can start $R'(k, n)$ two time-slices sooner. □

5 Reversal

We now give a construction reversing the contents of n wires in depth $2n + 2$. We then show that any such circuit has depth at least $2n + 1$.

5.1 Upper bound on reversal

Theorem 5.1. *We can reverse n wires in depth $2n + 2$. The circuit has size $n^2 - 1$.*

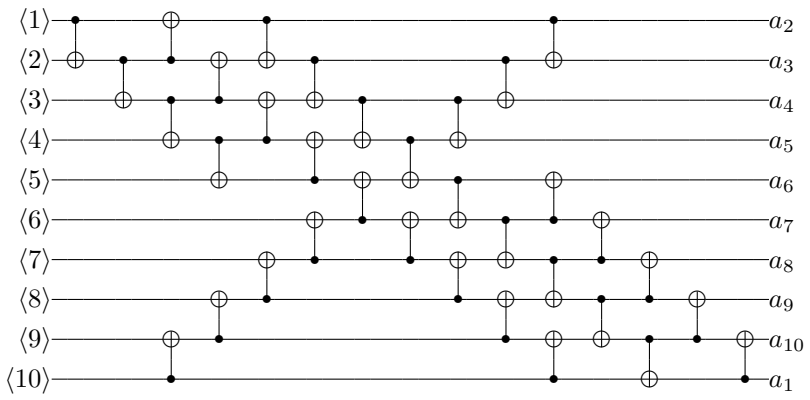

Figure 5: Rotation of 10 wires ($k = 5$) in depth 15.

for $i = \ell$ **to** $m - 1$:
    $\langle i+1 \rangle \oplus= \langle i \rangle$
for $i = \ell$ **to** $m - 1$:
    $\langle i \rangle \oplus= \langle i+1 \rangle$
for $i = \ell$ **to** $m - 1$:
    $\langle i+1 \rangle \oplus= \langle i \rangle$
for $i = m - 2$ **down to** ℓ:
    $\langle i+1 \rangle \oplus= \langle i \rangle$

Figure 6: Rotation circuit $R(\ell, m)$.

An example of this construction for $n = 9$ appears in Figure 7.

Proof. The reversal circuit is described in Figure 8. The subcircuit R_0 adds the contents of each wire with an even index into its neighbors; the subcircuit R_1 adds the odd-indexed wires into their neighbors. We alternate between these two operations.

For a given value of i, we will keep track of which wires depend on a_i over time. First suppose that i is even. To simplify matters, we will see what the effect of successive applications of R_0 and R_1 would be if there were wires corresponding to arbitrarily small and large integers. After the first application of R_0, since i is even, we perform $\langle i-1 \rangle \oplus= \langle i \rangle$ and $\langle i+1 \rangle \oplus= \langle i \rangle$, so a_i gets added to $\langle i-1 \rangle$ and $\langle i+1 \rangle$. Thus a_i affects $\langle i-1 \rangle$, $\langle i \rangle$, and $\langle i+1 \rangle$. After we next apply R_1, $\langle i-1 \rangle$ and $\langle i+1 \rangle$ are added to their neighboring wires, so a_i affects $\langle i-2 \rangle$ through $\langle i+2 \rangle$. (The effects of the two additions to $\langle i \rangle$ cancel.) In general, after applying R_0 and R_1 a total of t times, a_i will affect $\langle i-t \rangle$ through $\langle i+t \rangle$.

Now let us take into account the fact that we only have wires $\langle 1 \rangle$ through $\langle n \rangle$. During the ith application of an R-subcircuit, we cannot add $\langle 1 \rangle$ to $\langle 0 \rangle$, since the latter does not exist, so $\langle 1 \rangle$ is still the lowest-numbered wire affected by a_i. During the $(i+1)$th application of an R-subcircuit, $\langle 2 \rangle$ is added to $\langle 1 \rangle$, so $\langle 1 \rangle$ no longer depends on a_i, and $\langle 2 \rangle$ is now the first wire affected by a_i. Therefore, after t applications of R-subcircuits, for $t \geq i$, the lowest-numbered wire affected by a_i is $\langle t-i+1 \rangle$. Similarly, for $t > n-i$, the highest-numbered wire affected by a_i is $\langle n-(t-1-(n-i)) \rangle = \langle 2n-t-i+1 \rangle$. (We can see this by interchanging i and $n+1-i$,

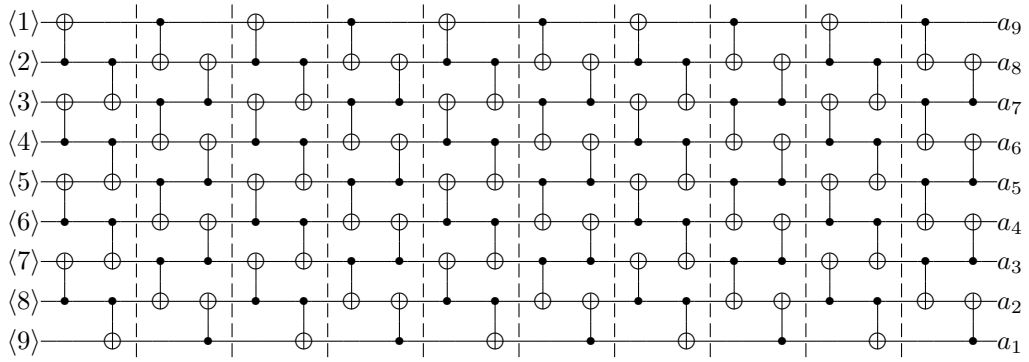

Figure 7: Reversal of 9 wires in depth 20.

R_0:
    for $i = 1$ **to** $\lfloor n/2 \rfloor$:
        $\langle 2i-1 \rangle \oplus= \langle 2i \rangle$
    for $i = 1$ **to** $\lfloor (n-1)/2 \rfloor$:
        $\langle 2i+1 \rangle \oplus= \langle 2i \rangle$
R_1:
    for $i = 1$ **to** $\lfloor n/2 \rfloor$:
        $\langle 2i \rangle \oplus= \langle 2i-1 \rangle$
    for $i = 1$ **to** $\lfloor (n-1)/2 \rfloor$:
        $\langle 2i \rangle \oplus= \langle 2i+1 \rangle$
Reversal:
    for $T = 0$ **to** n:
        if T is even: Apply R_0
        else: Apply R_1

Figure 8: Reversal circuit. For $n > 2$, the subcircuits R_0 and R_1 each have depth 2.

relabeling the wires in the opposite order, and interchanging R_0 and R_1 if n is even.) That is, for t bigger than both i and $n-i$ (and not too large), a_i will affect exactly the wires $\langle t-i+1 \rangle$ through $\langle 2n-t-i+1 \rangle$ after t applications of R-subcircuits.

Our circuit applies R-subcircuits a total of $n+1$ times. After n of these, a_i affects exactly wires $\langle n+1-i \rangle$ through $\langle n+1-i \rangle$; that is, a_i affects only $\langle n+1-i \rangle$. Since this nth application writes to wires of the opposite parity of $\langle n+1-i \rangle$, the $(n+1)$th application will write to wires of the same parity as $\langle n+1-i \rangle$, and $\langle n+1-i \rangle$ will still be the only wire affected by a_i.

Finally, we consider the case with i odd. After the first application of an R-subcircuit, $\langle i \rangle$ is still the only wire affected by a_i. Then, as above, after n more applications, $\langle n+1-i \rangle$ is the sole wire affected by a_i.

We have shown that, after our circuit runs, the wire $\langle n+1-i \rangle$ will depend on a_i, but no other wire $\langle n+1-j \rangle$ for $j \neq i$ will. Turning this around, we see that the final value of $\langle n+1-i \rangle$ does not depend on a_j for $j \neq i$, so that this final value must, in fact, be equal to a_i. We have performed reversal, as desired. □

For $n = 2$, the subcircuits R_0 and R_1 each have depth 1, so the overall depth of our reversal circuit is 3. For $n > 2$, the depth is $2n+2$.

5.2 Lower bound on reversal

For $2 \leq n \leq 6$, computer searches confirm that the above construction is optimal. We conjecture that the depth of any circuit performing reversal for $n \geq 3$ is at least $2n+2$. We now show that any such circuit has depth at least $2n+1$.

Lemma 5.2. *For any $k \leq n/2$, any circuit reversing n wires contains at least $2k+1$ gates between wires $\langle k \rangle$ and $\langle k+1 \rangle$ and also between wires $\langle n-k \rangle$ and $\langle n-k+1 \rangle$. If k is not $n/2$, then there must be at least $2k+1$ such gates before the last time-slice.*

Proof. Let R be a circuit reversing $\langle 1 \rangle, \ldots, \langle n \rangle$. We show that R must have at least $2k+1$ gates between wires $\langle k \rangle$ and $\langle k+1 \rangle$; the proof for $\langle n-k \rangle$ and $\langle n-k+1 \rangle$ is analogous.

We write the contents of the wires at any given time as a block matrix

$$M = \begin{pmatrix} W & X \\ Y & Z \end{pmatrix}, \tag{1}$$

where W is $k \times k$, X is $k \times (n-k)$, Y is $(n-k) \times k$, and Z is $(n-k) \times (n-k)$. The matrix M changes as we apply R. Initially, W and Z are identity matrices of sizes k and $n-k$, and X and Y are 0. When we conclude, W, X, Y, and Z have ranks 0, k, k, and $n-2k$, respectively.

The ranks of W, X, Y, and Z are affected only by gates between wires $\langle k \rangle$ and $\langle k+1 \rangle$. Each upward gate $\langle k \rangle \oplus= \langle k+1 \rangle$ changes the ranks of W and Y by at most 1, and each downward gate $\langle k+1 \rangle \oplus= \langle k \rangle$ changes the ranks of X and Z by at most 1. Each of the four ranks has to change by k. We conclude that there are at least k upward and k downward gates in R.

Furthermore, suppose that the first gate between $\langle k \rangle$ and $\langle k+1 \rangle$ is upward. At this point X is still 0, so the gate cannot affect the rank of W; the circuit R requires k more upward gates. Similarly, if the first gate is downward, it cannot affect the rank of Z, and R requires k additional downward gates. Hence, there must be at least $2k+1$ gates between $\langle k \rangle$ and $\langle k+1 \rangle$.

Finally, if k is not exactly $n/2$, then any gate in the last time-slice cannot affect any of the ranks of W, X, Y, Z, so all of the gates accounted for above must occur in earlier time-slices. □

Theorem 5.3. *Reversing $n \geq 3$ wires requires depth at least $2n+1$ and size at least $\left\lfloor \frac{1}{2}n^2 \right\rfloor + n$.*

Proof. First, suppose $n = 2r$. Given any circuit R for reversal, we obtain another reversal circuit by vertically flipping the last time-slice of R (that is, conjugating it by reversal) and moving it to the beginning of the circuit. We may therefore assume, without loss of generality, that the last time-slice contains a gate between $\langle r+1 \rangle$ and $\langle r+2 \rangle$.

By Lemma 5.2, there are at least $2r+1$ gates between wires $\langle r \rangle$ and $\langle r+1 \rangle$ and at least $2(r-1)+1 = 2r-1$ gates between wires $\langle r+1 \rangle$ and $\langle r+2 \rangle$ before the last time-slice. Hence, there are at least $4r+1 = 2n+1$ gates involving $\langle r+1 \rangle$, giving the lower bound on depth. If we sum over all locations, we find that the total number of gates is at least

$$2r + 1 + 2\sum_{i=1}^{r-1}(2i+1) + 1 = \frac{n^2+2n}{2}.$$

Second, suppose $n = 2r+1$. Again, we may assume that the last time-slice contains a gate between $\langle r+1 \rangle$ and $\langle r+2 \rangle$. Now we have at least $2r+1$ gates between wires $\langle r \rangle$ and $\langle r+1 \rangle$ and at least $2r+2$ gates between $\langle r+1 \rangle$ and $\langle r+2 \rangle$. This gives a total of $4r+3 = 2n+1$ gates involving $\langle r+1 \rangle$, meaning we must have at least $2n+1$ time-slices. The total number of gates is at least

$$2\sum_{i=1}^{r}(2i+1) + 1 = \frac{n^2+2n-1}{2}.$$

□

6 Permutation

We now discuss the more general problem of permuting the n input bits. It is easier to visualize the problem by imagining that the wire $\langle i \rangle$ contains the data a_i with the attached label $\sigma(i)$. We then wish to sort the data by their labels. When we finish, the wire $\langle i \rangle$ will have the label i, and hence the bit $a_{\sigma^{-1}(i)}$, as desired.

Theorem 6.1. *For any $\sigma \in S_n$, there is a circuit implementing σ with depth at most $3n$ and size at most $3\binom{n}{2}$.*

Proof. It is convenient to pretend that our basic operation is a swap of two adjacent bits; we can implement such a swap using three of our standard gates. Initially, our labels are in the order $\sigma(1), \ldots, \sigma(n)$; after each swap, the order changes. When the circuit completes, we want the labels to be sorted.

To effect the swaps, we use an n-bit *sorting network*. The basic gate is a *conditional swap* on $\langle i \rangle$ and $\langle j \rangle$: if $i < j$ but the label on $\langle i \rangle$ is larger than the label on $\langle j \rangle$, then we swap the contents and labels of the two wires. A network of conditional swaps is a sorting network if, for any (valid) initial assignment of labels, at the end wire $\langle i \rangle$ has label i. We are interested in sorting networks using only conditional swaps on adjacent wires. See [3, Section 5.3.4] for more discussion.

Suppose we have a n-bit sorting network of depth d and size s, in which each conditional swap is between two adjacent wires. We will perform each swap only if the label of the second bit is less than that of the first bit. Since we know σ in advance, we know which swaps to leave in the network and which to leave out. The result will be a swap network with depth at most d and size at most s. The corresponding circuit has depth at most $3d$ and size at most $3s$.

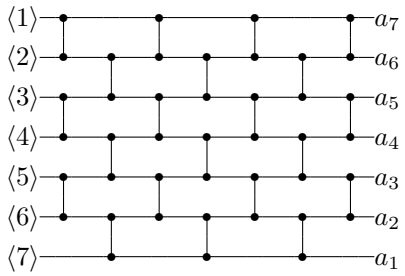

Figure 9: 7-wire sorting network in depth 7.

It merely remains to construct an efficient sorting network using only conditional swaps of adjacent wires. We use the odd–even transposition sort.[3] It has n steps, alternating between performing all conditional swaps of the form $(2j-1, 2j)$ and performing all conditional swaps of the form $(2j, 2j+1)$. An example with $n = 7$ is depicted in Figure 9. We have $s = n(n-1)/2$ and $d = n$ (unless $n = 2$, in which case $d = 1$). □

We observe that the above sorting network achieves the optimal d and s. First, note that each swap reduces the number of inversions by at most one. Since σ can have up to $\binom{n}{2}$ inversions, we must have $s \geq \binom{n}{2}$.

In addition, in an optimal sorting network, we cannot perform the same swap in consecutive time-slices. Thus, in any two consecutive time-slices, we can perform at most $n-1$ swaps. Hence, for all $n > 2$, we need at least $\lfloor n/2 \rfloor$ pairs of time-slices to accommodate $(n-1)\lfloor n/2 \rfloor$ gates, plus (at least) one more time-slice if n is odd. Hence, for all $n > 2$ we have $d \geq n$.

Clearly, for a particular permutation, we may be able to do better than Theorem 6.1 would suggest; see, for example, Sections 3, 4, and 5. A more difficult problem is determining the minimum depth for the worst possible σ.

[3] See Knuth [3, Exercise 5.3.4.37] for a proof of correctness and a brief history.

For $n \leq 6$, reversal is at least as hard as any other permutation: we can implement any permutation in depth $2n + 2$. We do not know whether this pattern holds for larger n.

7 Arbitrary Matrices

As noted in the Introduction, any circuit on n wires made up of CNOT gates computes a matrix in $\mathrm{GL}_n(2)$. Conversely, given a matrix, it is straightforward to build a circuit with depth $O(n^2)$.

More concretely, we suppose the initial state of the wires is described by the identity matrix I; each wire $\langle i \rangle$ contains the basis vector $\mathbf{e}_i$. If a circuit C applied to this initial state I results in state M, we say that C performs the transformation M. This map from circuits to matrices is a homomorphism.

The problems of building a circuit performing M and a circuit performing M^{-1}, for an arbitrary invertible matrix M, are equivalent. Notationally, we find the latter more convenient. Instead of building a circuit to perform M, we suppose the wires start in state M, and we construct a circuit to "undo" M and restore I. The reverse of this circuit will perform M.

In this section we give a constructive proof of the following result:

Theorem 7.1. *Let M be a matrix in $\mathrm{GL}_n(2)$. Then there is a circuit computing M with depth at most $5n$.*

Our construction uses the concept of a "northwest"-triangular matrix.

Definition 7.2. An $n \times n$ matrix M is *northwest-triangular* if $M_{ij} = 0$ for all $i + j > n + 1$.

We discuss the building blocks of our circuit in Section 7.1. In Sections 7.2 and 7.3, we prove the following propositions:

Proposition 7.3. *Let M be in $\mathrm{GL}_n(2)$. Given an n-wire sorting network of depth d, we can construct a circuit C of depth $2d$ such that MC is northwest-triangular.*

Proposition 7.4. *Let N be an invertible northwest-triangular matrix. Given an n-wire sorting network of depth d, we can construct a circuit R of depth $3d$ with $NR = I$.*

Proof of Theorem 7.1. Let M be any matrix in $\mathrm{GL}_n(2)$. By Proposition 7.3, using the odd-even transposition network of depth n, there is a circuit C of depth $2n$ such that MC is northwest-triangular. By Proposition 7.4 (using the same network), there is a circuit R of depth $3n$ with $MCR = I$. Then $R^{-1}C^{-1}$ computes M. □

The maximum possible size (that is, number of gates) of a depth-d circuit is $d\lfloor n/2 \rfloor$. The *density* of a circuit is its size divided by this maximum. The construction of Theorem 7.1 has size about $\frac{5}{2}n^2$ and density 1. We also have a construction with size about $2n^2$ and density 1/2. (See Section 8.) Note that, if we could construct a circuit with size $2n^2$ and density 1, we would have a solution with depth $4n$. We discuss this, and other reasons why we conjecture that circuits of depth $4n + O(1)$ may be possible, in Section 8.

7.1 Boxes

The building blocks for our circuits will be not individual CNOT gates, but *boxes*:

Definition 7.5. A *box* is a subcircuit on two adjacent wires $\langle i \rangle$ and $\langle i+1 \rangle$.

Every box performs some operation in $\mathrm{GL}_2(2)$. If u and v are the contents of the two input wires to a box, then the two output wires contain distinct elements of $\{u, v, u \oplus v\}$. Some researchers (for example, [2]) compute the costs of quantum circuits by counting arbitrary 2-qubit interactions; in such a model, the box is the fundamental unit.

Table 1: Depth of implementing boxes with input u, v.

First Output	Second Output	Depth
u	v	0
u	$u \oplus v$	1
$u \oplus v$	v	1
$u \oplus v$	u	2
v	$u \oplus v$	2
v	u	3

The depth of a box depends on the two output vectors, as shown in Table 1. If we want to perform an arbitrary operation in $\mathrm{GL}_2(2)$, then the depth of our box could be as large as 3. However, if we only specify one of the two outputs, and allow the other output to take whichever value is more convenient, we see that we can make do with boxes of depth 2.

7.2 Clearing Networks

We now prove Proposition 7.3. We use a sorting network to build a system of depth-2 boxes to convert any matrix into northwest-triangular form.

Proof of Proposition 7.3. We first perform a lower-triangular basis change; this does not involve changing the contents of any wires, but merely describes them differently. We then construct a circuit.

Let $V = \mathbf{F}_2^n$ be the space containing our wires. We define a *lexicographic order* on V. For $u, v \in V$, we write $u \prec v$ if there exists k such that $u \cdot \mathbf{e}_k = 0$, $v \cdot \mathbf{e}_k = 1$, and, for all $j > k$, $u \cdot \mathbf{e}_j = v \cdot \mathbf{e}_j$.

For each i, let v_i be the lexicographically least element of $a_i \oplus \mathrm{span}\{a_j : i < j \leq n\}$. Note that this is a lower-triangular basis change: for each i, $a_i \in v_i \oplus \mathrm{span}\{v_j : i < j \leq n\}$.

For each i, let $\pi(i)$ be the smallest j such that $v_i \cdot \mathbf{e}_{n+1-j} = 1$. By construction, π is a permutation: for any $k < \ell$, we must have $v_k \prec v_k \oplus v_\ell$, and therefore $\pi(k) \neq \pi(\ell)$.

Let $w_j = v_{\pi^{-1}(j)}$. The w_j satisfy

$$w_j \in \mathbf{e}_{n+1-j} \oplus \mathrm{span}\{\mathbf{e}_k : 1 \leq k < n+1-j\}.$$

Attach to each wire $\langle i \rangle$ the label $\pi(i)$. We maintain the following invariant:

- If a wire has value $\sum \alpha_j w_j$, and k is the label on some lower-numbered wire, then $\alpha_k = 0$.

The invariant is true initially because $\sum \alpha_j w_j = \sum \alpha_{\pi(i)} v_i$ and the basis change is lower-triangular. If we sort the labels and maintain the invariant, then when we are done, the value of wire $\langle i \rangle$ is in

$$w_i \oplus \operatorname{span}\{w_j : i < j \leq n\} = \mathbf{e}_{n+1-i} \oplus \operatorname{span}\{\mathbf{e}_j : 1 \leq j < n+1-i\},$$

so the wires specify a northwest-triangular matrix.

We now build a circuit C that sorts the labels while maintaining the invariant. We start with a sorting network S of depth d and replace each conditional swap in S by a box. Suppose we have two inputs to a box: $\langle i \rangle$ has value u and label j, and $\langle i+1 \rangle$ has value v and label k. If $j < k$, we do nothing. If $j > k$, we swap the two labels, and we also perform a box as described below. When the network concludes, we will have sorted the labels, as desired.

Let W be the span of all w_ℓ for $\ell \neq k$. The space W has codimension 1, so at least one of $\{u, v, u \oplus v\}$ lies in W. We can perform a box on $\langle i \rangle$ and $\langle i+1 \rangle$ that writes a vector in W to wire $\langle i+1 \rangle$. This maintains the invariant for wires $\langle i \rangle$ and $\langle i+1 \rangle$, as desired, and other wires are unaffected.

Each box in C comes from a conditional swap in S. We are specifying only one output of each box, so each box has depth at most 2. Hence, the depth of C is at most $2d$. □

7.3 Reversal Networks

We now prove Proposition 7.4: we reduce any northwest-triangular matrix to the identity. As before, we use a sorting network to build a system of boxes. However, in this case our boxes are permitted to have depth 3.

Proof of Proposition 7.4. We first label each input wire $\langle i \rangle$ with $n+1-i$. We take a sorting network S of depth d and convert S to a reversal network; we include exactly those conditional swaps that are used when input wire $\langle i \rangle$ has label $n+1-i$. (The new network will have size $\binom{n}{2}$; if S has the minimal size $\binom{n}{2}$, then it already is a reversal network.) We make each remaining swap unconditional: we definitely swap the two labels.

Consider a swap between $\langle i \rangle$, with value u and label k, and $\langle i+1 \rangle$, with value v and label j. Note that $k > j$. If $u \cdot \mathbf{e}_j = 0$, we replace the swap with a depth-3 box exchanging u and v. If $u \cdot \mathbf{e}_j = 1$, then we replace the swap with the depth-2 box that first adds u into v and then adds $u \oplus v$ into u; this has the effect of replacing u by $u \oplus v$ and then exchanging (the new) u and v.

We claim that this circuit maintains the following invariants:

1. If u is on the wire with label k, then $u \cdot \mathbf{e}_k = 1$ and $u \cdot \mathbf{e}_\ell = 0$ for $\ell > k$.
2. If u is on $\langle i \rangle$ with label k, and $\langle h \rangle$ has label j, with $h < i$ and $j < k$, then $u \cdot \mathbf{e}_j = 0$.

Initially, $\langle i \rangle$ has label $n+1-i$. The first invariant holds because N is an invertible northwest-triangular matrix. The second invariant holds vacuously, as there are no such pairs of wires.

What is the effect of a single box between wires $\langle i \rangle$ and $\langle i+1 \rangle$ with values u and v and labels k and j? The box necessarily maintains the first invariant. Swapping u and v has no effect. The step replacing u by $u \oplus v$ also is not a problem: $k > j$ implies $v \cdot \mathbf{e}_\ell = 0$ for all $\ell \geq k$.

This circuit also maintains the second invariant. It clearly still holds for all wires besides $\langle i \rangle$ and $\langle i+1 \rangle$. The value v and label j move unchanged from wire $\langle i+1 \rangle$ to wire $\langle i \rangle$, so it holds for

$\langle i \rangle$ as well. If label ℓ is on wire $\langle h \rangle$, with $h < i$ and $\ell < k$, then $u \cdot \mathbf{e}_\ell = 0$. Also, $v \cdot \mathbf{e}_\ell = 0$, either by the first invariant if $\ell > j$, or by the second if $\ell < j$, so $(u \oplus v) \cdot \mathbf{e}_\ell = 0$. Finally, we have designed the box so that the output value on $\langle i+1 \rangle$, either u or $u \oplus v$, is orthogonal to $\mathbf{e}_j$.

When R concludes, the labels are in order; wire $\langle i \rangle$ has label i. The two invariants then imply that $\langle i \rangle$ contains $\mathbf{e}_i$; that is, we have reached the identity matrix. □

7.4 Lower Bounds

By Theorem 5.3, reversal requires depth $2n+1$. Hence, we have already shown that the minimum depth for the worst-case matrix in $\mathrm{GL}_n(2)$ is at least $2n+1$. We now argue that almost all invertible matrices require about this depth. By "almost all invertible matrices" we mean a proportion of elements of $\mathrm{GL}_n(2)$ tending to 1 as n goes to ∞. First we quote a well-known result on ranks of random matrices.

Theorem 7.6. *As n goes to ∞, the proportion of $n \times n$ matrices over $\mathbf{F}_2$ having rank at most $n-c$ is $O(2^{-c^2})$.*

Sketch of proof. This follows from the fact that the number of $n \times n$ matrices of rank k is equal to the square of the number of $n \times k$ matrices of rank k divided by the number of invertible $k \times k$ matrices. To count these numbers of matrices, we use a standard formula of Landsberg [5]; see Stanley [6, Section 1.3] for a more recent exposition. □

Lemma 7.7. *Let $\epsilon > 0$ be given. For almost all matrices M in $\mathrm{GL}_n(2)$, every circuit implementing M has, for each integer k with $1 \le k \le n/2$, at least $2k - \epsilon n$ gates between wires $\langle k \rangle$ and $\langle k+1 \rangle$ and also between wires $\langle n-k \rangle$ and $\langle n-k+1 \rangle$.*

Proof. The proof uses the same technique as that of Lemma 5.2. As before, we consider wires $\langle k \rangle$ and $\langle k+1 \rangle$. Choose $M \in \mathrm{GL}_n(2)$ uniformly at random, and consider a circuit implementing M. We write the contents of the wires at any time as a block matrix, as in (1). Initially, X and Y are 0, and at the conclusion of the circuit, X and Y are two blocks of our matrix M. For large enough values of n and for a random choice of M, we expect X and Y each to have rank at least $k - (\epsilon/2)n$; since each gate between $\langle k \rangle$ and $\langle k+1 \rangle$ changes the total rank of X and Y by at most 1, we have at least $2k - \epsilon n$ such gates. □

Counting gates between different pairs of bits yields the following theorem:

Theorem 7.8. *Let $\epsilon > 0$ be given. For almost all matrices M in $\mathrm{GL}_n(2)$ every circuit implementing M requires depth at least $(2-\epsilon)n$ and size at least $(1-\epsilon)n^2$.*

A more careful analysis shows that the proportion of matrices in $\mathrm{GL}_n(2)$ that can be implemented in depth at most $2n-m$ is $O(2^{-(m-2)^2/8})$.

8 Open Questions

Let the "depth" of a matrix be the minimum depth of any circuit implementing the matrix. We have shown that the maximum depth of a matrix in $\mathrm{GL}_n(2)$ lies between $2n+1$ and $5n$. A natural question is whether we can close this gap.

For several reasons, the authors feel that the maximum depth may be only $4n + O(1)$. First, we consider circuit size: By Theorem 5.3, reversal requires at least $n^2/2$ gates, and the construction of Section 7 computes any matrix in at most $5n^2/2$ gates. Bob Beals [1] has shown that one can compute any matrix in only $2n^2$ gates. If we could pack these gates into a rectangular array, we could implement the matrix in depth $4n$.

More precisely, let ∇ be the set of all matrices implementable as ∇-shaped arrays of $\binom{n}{2}$ depth-2 boxes. A circuit in ∇ has size at most n^2 and depth at most $4n$. Beals showed [1] that $\nabla^2 = \mathrm{GL}_n(2)$: given M, he builds two circuits, one on either side of M, so that the product is the identity. Let Ξ be the set of all rectangular arrays of $\binom{n}{2}$ depth-2 boxes; a circuit in Ξ has size at most n^2 and depth at most $2n$. If we could similarly construct circuits in Ξ on either side of a matrix M to reduce it to the identity, then Ξ^2 would equal $\mathrm{GL}_n(2)$, and we could implement any matrix in depth $4n$.

By Proposition 7.3, we can use Ξ to reduce any matrix to northwest-triangular form. It is interesting to note that the subgroup of upper (or lower) triangular matrices in $\mathrm{GL}_n(2)$ has index $\prod_{i=1}^{n}(2^i - 1)$, but its order is only $\prod_{i=1}^{n} 2^{i-1} = 2^{(n^2-n)/2}$. Thus, one could argue that we are "working harder" to reduce a general matrix to northwest triangular form (in depth $2n$) than to reduce the triangular matrix to the identity (in depth $3n$). One could imagine that the latter reduction should be possible in the same depth as the former, providing further evidence that Ξ^2 might contain all of $\mathrm{GL}_n(2)$.

We performed exhaustive computer experiments for n up to 6. The maximum depths are shown in Table 2. While we are reluctant to draw inferences from such limited data, these values suggest that the maximum depth may be as small as $2n + O(1)$. In other words, the lower bound of Theorems 5.3 and 7.8 may be tight up to an additive constant.

Table 2: Maximum depth of a matrix in $\mathrm{GL}_n(2)$ for $n \leq 6$ obtained by exhaustive search.

n	2	3	4	5	6
depth	3	8	10	13	14

We also give some additional open questions:

- For addition, swap, and rotation, we have an upper bound for depth of $n + O(1)$ and a lower bound of $n - 1$. What is the correct additive constant?
- For $n \geq 3$, the optimal depth for reversal is either $2n + 1$ or $2n + 2$. Which is correct?
- What is the correct depth for a general permutation? For small n, reversal is at least as hard as any other permutation; does this hold for all n?
- For general matrices, we have a lower bound on size of $n^2/2$ and an upper bound of $2n^2$ [1]. What is the correct answer?
- As noted earlier, some researchers [2, 4] use the box as the basic unit of computation; in this model, we have a lower bound on depth of $n + 1$ (for reversal) and an upper bound of $2n$. What is the correct answer?
- Are there other classes of operations that can be implemented efficiently in this model?

The last question above is perhaps the most intriguing. Our focus was on selecting natural operations on n wires and then determining their depth. An alternative approach would be to consider all circuits of a given depth and see what other useful operations can be performed. Such an analysis might suggest new efficient circuits for arbitrary matrices and might even yield new approaches to quantum circuit design.

Acknowledgments

Tom Draper helped with our early work on addition, swap, and rotation. Bob Beals made many helpful suggestions and pointed out the generalization from Theorem 5.3 to Theorem 7.8.

References

[1] Robert M. Beals. Private communication, 2004.

[2] Austin G. Fowler, Simon J. Devitt, and Lloyd C. L. Hollenberg. Implementation of Shor's algorithm on a linear nearest neighbour qubit array. *Quantum Information and Computation*, 4(4):237–251, 2004.

[3] Donald E. Knuth. *Sorting and Searching*, volume 3 of *The Art of Computer Programming*. Addison–Wesley, second edition, 1998.

[4] Samuel A. Kutin. Shor's algorithm on a nearest-neighbor machine. quant-ph/0609001, 2006.

[5] G. Landsberg. Über eine Anzahlbestimmung und eine damit zusammenhägende Reihe. *J. Reine Angew. Math*, 111:87–88, 1893.

[6] Richard P. Stanley. *Enumerative Combinatorics*, volume 1. Cambridge University Press, 1997.

[7] Rodney Van Meter. *Architecture of a quantum multicomputer optimized for Shor's factoring algorithm*. PhD thesis, Keio University, 2006. Also quant-ph/0607065.

[8] Rodney Van Meter and Kohei Itoh. Fast quantum modular exponentiation. *Physics Review Letters A*, 71:052320, 2005.