PhD Project Report
Optimal Designs for Fault-Tolerant Quantum Computers

Author: Luke Heyfron

Supervisor: Dr. Earl T. Campbell

June 2017

1 Introduction

Certain high threshold quantum error correction codes such as the toric code have transversal implementations of quantum logic gates that generate the logical Clifford group. However, the Clifford group in isolation is insufficient for universal quantum computation: one quantum gate outside the Clifford group must be included in the gate set. This is often chosen to be the T gate, which unfortunately does not have a transversal implementation on the toric code. A standard way to apply a T gate fault-tolerantly to a logically encoded qubit is to prepare a high-fidelity ancillary T-type magic state using magic state distillation protocols, then apply a teleportation gadget between the ancilla and the target qubit. This method involves many more elementary gates than the transversal Clifford gates, which motivates the cost model for logical quantum gates where the Clifford gates are "free" and T gates are "expensive". Consequentially, it is of interest to find ways to reduce the number of T gates used in implementations of quantum circuits.

Previous work in this area.

In this report an algorithm is presented for reducing the T count for n-qubit quantum circuits composed of CNOT and T gates. It is shown how it is possible to generalize this algorithm for the universal gate set $\{H, CNOT, T\}$. The rest of the paper is structured as follows: section 2 we define the terminology and notation conventions used for the rest of the paper; section 3 we define the problem in terms of quantum circuits and operators as well as in terms of a special kind of matrix representation, which facilitates the description of our solution. In section 4 we describe the algorithm presented in this paper for reducing the T count; in section 5 we present the results of simulations of this algorithm compared to previous algorithms for both randomized circuits over CNOT and T as well as practical circuits of a universal gate set. Finally, we consolidate our results and discuss future work in section 6.

2 Theory and Terminology

2.1 Quantum Theory

We define the set of n-qubit computational basis states to be $\{|x\rangle\}$ for all $x \in \{0,1\}^n$, the set of binary tuples of length n.

The single qubit Pauli operators are defined in the computational basis as follows,

$$\sigma_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$
$$\sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{1}$$

The n-qubit Pauli group $\mathcal{P}^n$ is defined as the n-fold tensor product of single qubit Pauli operators, along with multiplicative factors of ± 1 and $\pm i$ i.e. $\mathcal{P}^n = \{(-1)^a \, i^b \sigma_c \mid a, b \in \{0,1\}, c \in \{0, x, y, z\}\}^n$.

The operators CNOT, H and T are given by

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \tag{2}$$

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \tag{3}$$

$$T = \begin{pmatrix} 1 & 0 \\ 0 & \omega \end{pmatrix}, \tag{4}$$

where we define $\omega = e^{i\frac{\pi}{4}}$. A single qubit operator G applied to the i^{th} qubit of an n-qubit system is expressed as $G^{(i)}$. A controlled-G operator where G is a single qubit target operator is expressed as control-$G^{(t,c)}$ where the t^{th} and c^{th} qubits are the target and control qubit, respectively.

The n-qubit Clifford group $\mathcal{C}^n$ is defined as the set of operators that maps $\mathcal{P}^n$ to itself under conjugation, $\mathcal{C}^n = \{U \mid UPU^\dagger = P', P, P' \in \mathcal{P}^n\}$.

The k^{th} level of the Clifford hierarchy on n-qubits, $\mathcal{C}_k^n$, is defined inductively,

$$\mathcal{C}_k^n = \{U \mid UPU^\dagger = Q, P \in \mathcal{P}^n, Q \in \mathcal{C}_{k-1}^n\}, \tag{5}$$

with $\mathcal{C}_1^n = \mathcal{P}^n$.

We define $\mathcal{D}_k^n \subseteq \mathcal{C}_k^n$ as the set of diagonal elements of the k^{th} level of the Clifford hierarchy.

2.2 Classical Coding Theory

2.3 Reed-Muller Codes

3 The Gate Synthesis Problem

NOTES from Earl's paper: - magic states model formalized by Bravyi and Kitaev - Clifford operations are natively protected against noise in many error correcting codes including 2D topological codes such surface codes and 2D color codes. - Clifford operations: preparation of —0¿ states, measurement of Pauli group operators, Clifford group operators (normalizer of Pauli group), classical feed forward, access to classical random number generator. - Non-Clifford operations aren't natively protected so are costly. - Magic state distillation required (multiple layers) each of which involves many Clifford operations. Therefore, cost of non-clifford operations is significantly more than clifford operations - We measure resources by counting the number of logical T gates used in a circuit. - Using Campbell and Howard's synthillation protocol, the circuits can be implemented fault tolerantly with low overhead.

An n-qubit quantum circuit C of length k is a time-ordered set of operators taken from an elementary gate set G on n qubits. For instance, if C is such a quantum circuit, then $C = \{C_1, C_2, ..., C_k\}$ where $C_t \in G^n$ is the operator applied at the t^{th} time step.

If C is a quantum circuit, then C is said to implement the unitary U_C, which is given by

$$U_C = \prod_{t=k}^{1} C_t. \tag{6}$$

Two circuits are said to be equivalent if they implement the same unitary. I.e. $C \equiv_U C'$ if $U_C = U_{C'}$. We define $E(g)$ to be the cost function of applying a particular gate $g \in G$ to a logical qubit. The total cost of a circuit is therefore,

$$E(C) = \sum_{t=1}^{k} E(C_t). \tag{7}$$

Given some initial circuit C, the gate synthesis problem is to find some circuit C' such that $\min\{E(C') \mid U_{C'} = U_C\}$.

According to our cost model for quantum gates, $E(\text{Clifford}) = 0$ and $E(T) = 1$. The total cost function for a circuit C is therefore,

$$E(C) = \sum_{t=1}^{k} E(C_t) = \tau(C), \tag{8}$$

where we define $\tau(C)$ to be the number of T gates in circuit C, which means the gate synthesis problem is equivalent to minimizing the number of T gates. We define the T count of a unitary U as $\tau(U) = \min\{\tau(C) \mid U = U_C\}$.

3.1 Circuits Composed of CNOT and T Gates

Any unitary U generated by the CNOT and T gate can be decomposed as,

$$U = VW, \tag{9}$$

where V is a member of the Clifford group with $\tau(V) = 0$ and $W \in \mathcal{D}_3^n$. It follows that $\tau(W) = \tau(U)$. Due to V not contributing to the T count of U, we will assume that U is already diagonal to simplify the analysis.

The action of a unitary $U \in \mathcal{D}_3^n$ on a computational basis state can be written as follows,

$$U\left|x\right\rangle = \omega^{f_U(x)}\left|x\right\rangle, \tag{10}$$

where $f_U(x)$ is known as the *phase function* for U. The phase function can be determined by tracking the state of each qubit through the circuit, updating appropriately after each CNOT gate, and adding a term after each T gate. This results in the general form for a phase function,

$$f_U(x) = \sum_{y \in \{0,1\}^n \setminus 0} \mathbf{a}_y \bigoplus_{i=1}^{n} x_i^{y_i} \mod 8, \tag{11}$$

where $\mathbf{a} \in \mathbb{Z}_8^{2^n}$ is called an *implementation vector*. Modulo 8 arithmetic is due to ω being the eight root of unity. This decomposition of the phase function is known as the *phase polynomial* decomposition. Once an implementation vector for U is found, it is possible to construct a circuit C that implements U using $\tau(C) = |\mathbf{a} \mod 2|$ T gates. There exist multiple implementation vectors that give rise to the same phase functions. This is due to the existence of non-trivial implementation vectors that implement the all-zero phase function.

Definition 3.1. Gate Synthesis Matrix. Let $\mathbf{a}$ be an implementation vector for a phase function $f(x)$. A gate synthesis matrix is an $n \times m$ matrix, A, whose elements $A_{i,j} \in GF(2)$, where for each y satisfying $(\mathbf{a}_y \mod 2) = 1$, there is one column in A such that $A_{i,j} = y_i$ for some $j \in [1, m]$.

Note from the definition of a gate synthesis matrix, $m = |\mathbf{a} \mod 2|$, and that any two gate synthesis matrices that differ only by column permutations should be considered equivalent.

Problem 3.1. *Gate Synthesis Problem (version 1). Let A be some gate synthesis matrix.*

4 Solutions to the Gate Synthesis Problem

4.1 The Lempel Algorithm

4.2 The Extended Lempel Algorithm

Definition 4.1. *Signature tensor.* Let A be an $n \times m$ gate synthesis matrix. The signature tensor of A is an $n \times n \times n$ symmetric tensor on GF(2) defined as,

$$(S(A))_{\alpha,\beta,\gamma} = \sum_{j=1}^{m} A_{\alpha,j} A_{\beta,j} A_{\gamma,j}. \tag{12}$$

Note that as $S(A)$ is symmetric, it is invariant under any permutation of indices. Hence, we shall omit any repeated index for the sake of brevity, e.g. $S_{\alpha,\alpha,\beta} = S_{\alpha,\beta}$.

Let $\mathcal{I} = \{(\alpha, \beta, \gamma) \mid 1 \leq \alpha \neq \beta \neq \gamma \leq n\}$ be the set of all 3-tuples such that each element falls in the range $[1, n]$ and is unique. We define $\chi(A, x)$ to be an $|\mathcal{I}| \times m$ matrix that is a function of A and x, a $n \times m$ matrix and a column vector of length n, respectively:

$$(\chi(A, x))_{i,j} = x_\alpha x_\beta A_{\gamma,j} + x_\beta x_\gamma A_{\alpha,j} + x_\gamma x_\alpha A_{\beta,j}, \tag{13}$$

where (α, β, γ) is the i^{th} element of $\mathcal{I}$.

Lemma 4.1. *Let A and $A' = A + xy^T$ be two gate synthesis matrices where x, y are arbitrary column vectors of dimension n and m, respectively. $S(A) = S(A')$ if all of the following conditions are met:*

1. $|y| = 0$
2. $Ay = 0$
3. $\chi(A,x)\, y = 0$.

Proof. We begin the proof by finding an expression for $S(A')$ using equation 12,

$$(S(A'))_{\alpha,\beta,\gamma} = \sum_{j=1}^{m} \left(A_{\alpha,j} + x_\alpha y_j\right)\left(A_{\beta,j} + x_\beta y_j\right)\left(A_{\gamma,j} + x_\gamma y_j\right), \tag{14}$$

and expanding the brackets,

$$\begin{aligned}(S(A'))_{\alpha,\beta,\gamma} = \sum_{j=1}^{m}(&A_{\alpha,j}A_{\beta,j}A_{\gamma,j} + x_\alpha x_\beta x_\gamma y_j \\ &+ x_\alpha x_\beta A_{\gamma,j} y_j + x_\beta x_\gamma A_{\alpha,j} y_j + x_\gamma x_\alpha A_{\beta,j} y_j \\ &+ x_\alpha A_{\beta,j} A_{\gamma,j} y_j + x_\beta A_{\gamma,j} A_{\alpha,j} y_j + x_\gamma A_{\alpha,j} A_{\beta,j} y_j).\end{aligned} \tag{15}$$

If we sum the first term over all j, we find that it becomes equal to $S(A)$. The task is to show that the remaining terms sum to zero under the specified conditions. Next, we use the definitions of $|y|$, Ay and $\chi(A,x)\, y$ to simplify the result,

$$(S(A'))_{\alpha,\beta,\gamma} = (S(A))_{\alpha,\beta,\gamma} + x_\alpha x_\beta x_\gamma |y| + x_\alpha x_\beta \left(Ay\right)_\gamma + x_\beta x_\gamma \left(Ay\right)_\alpha + x_\gamma x_\alpha \left(Ay\right)_\beta + \left(\chi(A,x)\; y\right)_i, \tag{16}$$

where we define i such that (α,β,γ) is the ith element of $\mathcal{I}$.

By applying condition (1), the second term is eliminated; by applying condition (2), the next three terms are eliminated, and by applying condition (3), the final term is eliminated. □

Lemma 4.2. *Let A be an $n \times m$ gate synthesis matrix where all columns are unique and non-zero. Let $A' = A + xy^T$ where x and y are column vectors on GF(2) of length n and m, respectively, defined such that $x_i = A_{i,a} + A_{i,b}$ for some $a, b \in [1, m]$ and $y_a + y_b = 1$. The columns a and b of A' are identical.*

Proof. We begin the proof by finding expressions for the matrix elements of A' in terms of A, x and y,

$$A'_{i,j} = A_{i,j} + x_i y_j, \tag{17}$$

and substitute the definition of x,

$$A'_{i,j} = A_{i,j} + (A_{i,a} + A_{i,b}) y_j. \tag{18}$$

Now we can find the elements of the columns a and b of A',

$$A'_{i,a} = A_{i,a} + (A_{i,a} + A_{i,b}) y_a, \tag{19}$$

$$A'_{i,b} = A_{i,b} + (A_{i,a} + A_{i,b}) y_b. \tag{20}$$

We substitute in the condition $y_b = y_a + 1$ into 20,

$$\begin{aligned}A'_{i,b} &= A_{i,b} + (A_{i,a} + A_{i,b})(y_a + 1) \\ &= A_{i,a} + (A_{i,a} + A_{i,b}) y_a \\ &= A'_{i,a}.\end{aligned} \tag{21}$$

□

NOTE: Before defining signature tensors, establish the notion that the signature tensor is a unique representation for an equivalence class of phase functions that are equal to each other up to a Clifford group operator. This can be done by first proving that all phase functions that have the same weighted polynomial are Clifford equivalent. All gate synthesis matrices that have the same signature tensor have the same weighted polynomial, therefore are Clifford equivalent.

Algorithm 1 Extended Lempel Algorithm (Base)

Input: A matrix A with n rows and m columns whose elements are members of $\mathbb{Z}_2$.
Output: A matrix A' with n rows and $p = m - q$ columns such that $a \in [0, 2]$ and $S(A') = S(A)$.

- Let $\mathrm{col}_j(A)$ be a function that returns the j^{th} column of A.
- Let $\mathrm{cols}(A)$ be a function that returns the number of columns of A.
- Let $\mathrm{nullspace}(A)$ be a function that returns a matrix whose columns generate the right nullspace of A.

```
procedure LempelXbase
    Initialize $A' \leftarrow A$
    for all $1 \le a < b \le m$ do
        $x \leftarrow \mathrm{col}_a(A) + \mathrm{col}_b(A)$
        $\tilde{A} \leftarrow \begin{pmatrix} A \\ \chi(A, x) \end{pmatrix}$
        $N \leftarrow \mathrm{nullspace}(\tilde{A})$
        for all $1 \le k \le \mathrm{cols}(N)$ do
            $y = \mathrm{col}_k(N)$
            if $y_a + y_b = 1$ then
                if $|y| = 1 \pmod 2$ then
                    $A' \leftarrow \begin{pmatrix} A' & \mathbf{0} \end{pmatrix}$
                    $y \leftarrow \begin{pmatrix} y \\ 1 \end{pmatrix}$
                $A' \leftarrow A' + xy^T$
                Remove columns $a$ and $b$ from $A'$
                exit LempelXbase
```

Algorithm 2 Extended Lempel Algorithm (Full)

Input: A matrix A with n rows and m columns whose elements are members of $\mathbb{Z}_2$.
Output: A matrix A' with n rows and $p \le m$ columns such that p is minimal with respect to algorithm 1 and $S(A') = S(A)$.

```
procedure LempelX
start:
    $A' \leftarrow \mathrm{LempelXbase}(A)$
    if $A' = A$ then
        exit LempelX
    else
        goto start
```

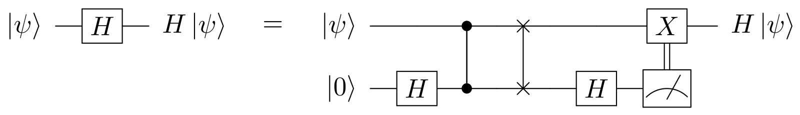

Figure 1: This circuit identity is used to trade the internal Hadamards of a circuit for external Hadamards, along with a Pauli-X correction conditioned on the measurement result of an ancilla.

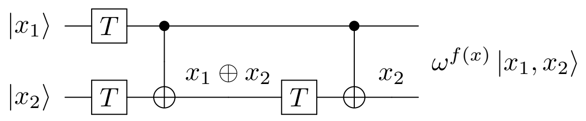

Figure 2: For circuits composed of CNOT and T gates, the phase polynomial is determined by updating the state of the target qubit of each CNOT gate and adding the appropriate term after each T gate. This circuit has phase polynomial $f(x) = x_1 + x_2 + x_1 \oplus x_2$.

NOTE: Then the full algorithm where we search over all column pairs i,j, form chi matrix and find nullspace basis vector y that yi + yj = 1 and append a column when —y— = 1. Exit condition is that no column pair is eliminated for one exhaustive search through pairs.

NOTE: Make part of the definition of a "lean" gate synthesis matrix that all columns are unique and non-zero.

NOTE: Establish somewhere that for both Lempel algorithms all addition is modulo 2 except where explicitly stated.

NOTE: use mathbf for all matrices and vectors

NOTE: The first part of the section shows that we can add an arbitrary column vector x to an arbitrary subset of the columns of a gate synthesis matrix given by y without altering the signature tensor. The second part shows that choosing a specific value of x results in a new gate synthesis matrix that has two identical columns, which can subsequently be removed.

NOTE: Define clifford equivalence and gate synthesis matrices like in earl's paper. Say something like "throughout we work in the framework of gate synthesis matrices as defined in ¡cite earl¿" then summarise the notation used.

NOTE: In this section we will show how to reduce the width of a gate synthesis matrix by 2, given certain conditions are met. Then one can simply apply this method iteratively until the conditions are no longer met.

5 Experimental Results

5.1 Random Circuits Composed of CNOT and T Gates

5.2 Universal Practical Circuits

6 Discussion and Conclusions

References

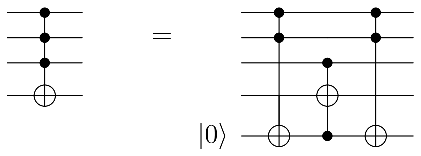

Figure 3: Circuit showing a method to construct the the 4-control Toffoli gate with standard Toffoli gates. The generalized n-controlled Toffoli can be constructed similarly with additional ancillas.