Notes for *How to efficiently select an arbitrary Clifford group element & Hadamard-free circuits expose the structure of the Clifford group*

Chengkai Zhu[1]

[1]*Institute for Quantum Computing, Baidu Research, Beijing 100193, China*

July 12, 2021

Abstract

In this note, we focus on a topic about how to generate a random uniformly distributed Clifford Operator. We will present the main content of the paper - *How to efficiently select an arbitrary Clifford group element* and *Hadamard-free circuits expose the structure of the Clifford group.* Robert Koenig and John A. Smolin, authors of the former paper propose an algorithm that has runtime $\mathcal{O}(n^3)$ to generate a random uniformly distributed Clifford Operator. And Sergey Bravyi and Dmitri Maslov, authors of the latter one, give an algorithm that has runtime $\mathcal{O}(n^2)$. We will have an inside look at how they do that in the following note and discuss about how we can decompose a Clifford into a circuit once we finish the former task.

1 Backgroud

1.1 Clifford Group

The Pauli group $\mathcal{P}_n$ on n qubits is generated by single-qubit Pauli operators

$$X_j=\left(\begin{array}{ll}0 & 1 \\ 1 & 0\end{array}\right), Y_j=\left(\begin{array}{cc}0 & -i \\ i & 0\end{array}\right), Z_j=\left(\begin{array}{cc}1 & 0 \\ 0 & -1\end{array}\right)$$

acting on the j th qubit, for $j=1,\ldots,n$. Consider the normalizer $\mathcal{N}\left(\mathcal{P}_n\right)=\left\{U \in U\left(2^n\right) \mid U \mathcal{P}_n U^{\dagger}=\mathcal{P}_n\right\}$ of $\mathcal{P}_n$ in the group of unitaries $U\left(2^n\right)$. The Clifford group $\mathcal{C}_n$ is this normalizer, neglecting the global phase: $\mathcal{C}_n=\mathcal{N}\left(\mathcal{P}_n\right)/U(1)$.

1.2 Problem of Generating a Random Uniformly Distributed Clifford Operator

We already have some approaches to generate a random uniformly unitary. However, the task becomes not that easy when the unitary we want is specifically a Clifford. The most straightforward way is to simply write down all elements of the Clifford Group and then pick randomly from the list. But it is impractical since the cardinality of the group

$$|\mathcal{C}_n| = 2^{n^2+2n} \prod_{j=1}^{n} \left(4^j - 1\right)$$

grows quickly with the number of qubits n.

Luckily, there are other methods that have been proposed: In [3] a method is given requiring time $\mathcal{O}(n^8)$ and producing an approximately random Clifford.

In 2014, authors in [1] used a map betweem *Symmetric Group* and *Clifford Group* to randomly pick Clifford group element by randomly picking a consecutive integer corresponding to a symmetric matrix. This method ends up with an algorithm that has runtime $\mathcal{O}(n^3)$, which gives a significant acceleration compared with the method in [3]. Another advantage of this method is that one can be inspired to use a symplectic matrix or even an interger to represent a Clifford Operator.

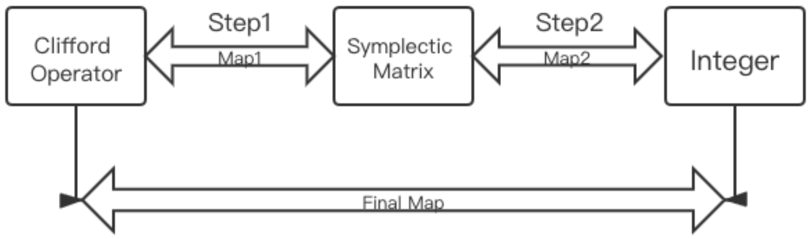
Years later, the authors in [2] gave an optimization of the problem from another aspect. They considered the structural properties of the Clifford group and present a new and simplier canonical form of any Clifford Operator, showing that any Clifford Ooperator can be uniquely written in the canonical form F_1HSF_2, where H is a layer of Hadamard gates, S is a permutation of qubits, and F_i are parameterized Hadamard-free circuits chosen from suitable subgroups of the Clifford group.

Now let's explain some details in these two papers respectively.

2 Details in ***How to efficiently select an arbitrary Clifford group element***

2.1 Main Idea

The main technique in this paper is giving a canonical map of consecutive integers to a Clifford group element. We will expain the details step by step as the processes showing bellow:



Notice any element $U \in \mathcal{C}_n$ is uniquely determined up to a global phase by its action by conjugation on the generators of $\mathcal{P}_n$, i.e. the list of parameters $(\alpha, \beta, \gamma, \delta, r, s)$ where

$\alpha, \beta, \gamma, \delta$ are $n \times n$ matrices of bits, and r, s are n -bit vectors defined by

$$U X_j U^{\dagger}=(-1)^{r_j} \prod_{i=1}^n X_i^{\alpha_{ji}} Z_i^{\beta_{ji}} \quad \text { and } \quad U Z_j U^{\dagger}=(-1)^{s_j} \prod_{i=1}^n X_i^{\gamma_{ji}} Z_i^{\delta_{ji}} \tag{1}$$

Note that because unitaries preserve commutation relations among the generators not all values for the matrices $\alpha, \beta, \gamma, \delta$ are allowed. This is what makes picking a random element of the group nontrivial. By the equations above, the task of drawing a random Clifford element can be rephrased as that of drawing from the corresponding distribution of parameters $(\alpha, \beta, \gamma, \delta)$ describing such an element.

2.2 Step 1

Interestingly, what we illustrate above are well-known properties of clifford. But what the authors of [1] did great is that they used that to come up with an idea of turning the problem of randomly picking Clifford into another one. The parameter $(\alpha, \beta, \gamma, \delta)$ has a lot to do with a symplectic matrix. Recall that

$$\mathcal{C}_n / \mathcal{P}_n \cong \operatorname{Sp}\left(2 n, \mathbb{F}_2\right) \equiv \operatorname{Sp}(2 n)$$

where the latter is the symplectic group on $\mathbb{F}_2^{2 n}$. If a representative $U \in \mathcal{C}_n / \mathcal{P}_n$ acts as (1), then the corresponding symplectic matrix S has entries:

$$\begin{aligned}
&\left(\alpha_{j 1}, \beta_{j 1}, \ldots, \alpha_{j n}, \beta_{j n}\right) \text { in column } 2 j-1 \text { and } \\
&\left(\gamma_{j 1}, \delta_{j 1}, \ldots, \gamma_{j n}, \delta_{j n}\right) \text { in column } 2 j, \text { for } j=1, \ldots, n
\end{aligned}$$

So the main lemma that the Algorithm in [1] based on is as follow:

Lemma 2.1. *Specifying an arbitrary element of the Clifford group is equivalent to specifying an element of the Pauli group and also an element from the symplectic group.*

Specifying an element of the Pauli group (up to an overall phase) simply requires picking the bitstrings r, s, which is trivial. Then the task is came down to specifying a symplectic matrix on $\mathbb{F}_2^{2 n}$ specifically.

2.3 Step 2

Novel part of Specifying a Symplectic Matrix:

Now we see there is a map between a Clifford Operator and a symplectic matrix. Then how it ended up with a map between intergers to a clifford operator? It is because another map and technique the authors use:

$$\begin{aligned}
\mathcal{S}_n & \rightarrow \operatorname{Sp}(2 n) / \operatorname{Sp}(2(n-1)) \\
(v, w) & \mapsto \quad\left[S_{v, w}\right]
\end{aligned}$$

You may have no clue what is going on. But the following paragraph will explain this clearly.

Basically, what the authors did to specify a symplectic matrix was called the *subgroup algorithm*. When G is a finite group with a nested chain of subgroups

$$G_1 \subset G_2 \subset \cdots \subset G_{n-1} \subset G_n = G$$

The map

$$\begin{aligned} G_n/G_{n-1} \times G_{n-1}/G_{n-2} \times \cdots \times G_2/G_1 \times G_1 &\to G \\ ([g_n], [g_{n-1}], \ldots, [g_2], g_1) &\to g_n g_{n-1} \cdots g_1 \end{aligned}$$

is an isomorphism. In particular, each $g \in G$ has a unique representation as $g_n g_{n-1} \cdots g_1$ with $[g_j] \in G_j/G_{j-1}$ for $j = 2, \ldots, n$ and $g_1 \in G_1$. This implies that given an element $g_j \in G_j$ representing a uniformly random coset $[g_j] \in G_j/G_{j-1}$ for every $j = 2, \ldots, n$, and a uniformly chosen random element $g_1 \in G_1$, we can obtain a uniformly distributed element of G by taking the product.

In our case we take $G_j = \operatorname{Sp}(2j)$ where the embedding $\operatorname{Sp}(2(j-1)) \to \operatorname{Sp}(2j)$ is given by $S \mapsto \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \oplus S$.

One thing worth mention is that we used to define a symplectic matrix use

$$\Lambda(n) = \begin{bmatrix} 0 & I_n \\ I_n & 0 \end{bmatrix}$$

when we say a symplectic matrix S satisfys $S\Lambda(n)S^T = \Lambda(n)$. However, the authors define as:

$$S\Lambda(n)S^T = \Lambda(n) \equiv \bigoplus_{i=1}^{n} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

The reason is that by doing so, the embedding of subgroup we mention before can be true!

Furthermore, there is a one-to-one correspondence between the set

$$\mathcal{S}_n := \left\{ (v, w) \in \mathbb{F}_2^{2n} \times \mathbb{F}_2^{2n} \mid \langle v, w \rangle = 1 \right\}$$

of symplectic pairs of vectors and the cosets $\operatorname{Sp}(2n)/\operatorname{Sp}(2(n-1))$. More precisely:

$$\begin{aligned} \mathcal{S}_n &\to \operatorname{Sp}(2n)/\operatorname{Sp}(2(n-1)) \\ (v, w) &\mapsto \quad [S_{v,w}] \end{aligned}$$

This is what we show at the beginning. And by randomly picking an interger we can obtain a desired pair (v, w).

2.4 Small Technique for Efficiency

Actually the work is pretty much done when we can build these maps and turn the problem of randomly picking a Clifford Operator to randomly picking an interger or intergers. But the authors did some improvement in the latter process in consideration of the efficiency of the algorithm.

After we get a symplectic pair $(v, w) \in \mathcal{S}_n$, we need an arbitrary basis for the space orthogonal to v and w. It is quite straightforward to use Symplectic Gram-Schmidt procedure and Gaussian elimination. However, the second technique used in the paper is to define another map called *transvection* in order to not using Gaussian elimination and makes the complexity of the algorithm from $\mathcal{O}(n^4)$ to $\mathcal{O}(n^3)$. In fact, it is well-known that $\mathrm{Sp}(2n)$ is generated by transvections.The map is as follows:

$$\begin{aligned} Z_h : \mathbb{F}_2^{2n} &\to \mathbb{F}_2^{2n} \\ v &\mapsto v + \langle v, h \rangle h \\ Z_h v =& v + \langle v, h \rangle h \end{aligned}$$

And there is a lemma to ensure this map can usefully work in the Algorithm.

Lemma 2.2. *Let $x, y \in \mathbb{F}_2^{2n} \backslash \{0\}$ be two non-zero vectors. Then*

$$y = Z_h x \qquad \text{for some } h \in \mathbb{F}_2^{2n}$$

or

$$y = Z_{h_1} Z_{h_2} x \qquad \text{for some } h_1, h_2 \in \mathbb{F}_2^{2n}$$

In other words, x can be mapped to y by at most two transvections.

Also, the authors give an algorithm that outputs either h or (h_1, h_2) satisfying equations above. Then we can see the whole Alogrithm in Appendix that runs in time $\mathcal{O}(n^3)$ after these details.

3 Details in ***Hadamard-free circuits expose the structure of the Clifford group***

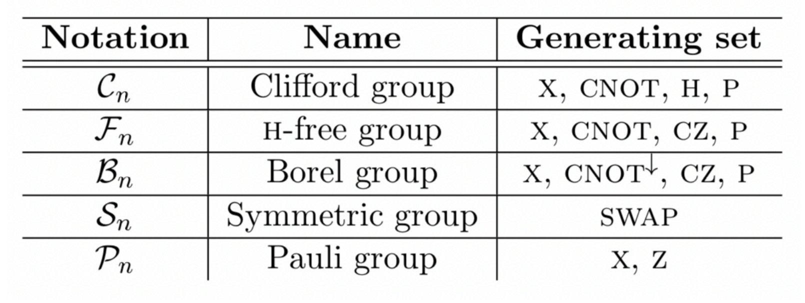
3.1 Main Idea

Recall that Koeing and Smolin's approach we explained above is to employ a sequence of symplectic transvections and maps to reduce the problem. However, in 2020, S. Bravyi and D. Maslov considered this problem more straightforwardly.

They show that any Clifford Operator can be uniquely written in the form $F_1 H S F_2$, where H is a layer of Hadamard gates, S is a permutation of n qubits, and F_i are Hadamard-free circuits chosen from suitable subgroups of the Clifford group, that depend on H and S. Also, they provide a simple explicit characterization of these subgroups and give a polynomial-time algorithm for computing all the layers in the above decomposition. Besides, the algorithm in this paper highlights a surprising connection between random uniform Clifford operators and the Mallows distribution on the symmetric group that plays an important role in several ranking algorithms.

3.2 Canonical Form of Clifford group

In order to give a canonical form of any clifford operator, one need some notions first:



Notation	Name	Generating set
$\mathcal{C}_n$	Clifford group	X, CNOT, H, P
$\mathcal{F}_n$	H-free group	X, CNOT, CZ, P
$\mathcal{B}_n$	Borel group	X, CNOT$^{\downarrow}$, CZ, P
$\mathcal{S}_n$	Symmetric group	SWAP
$\mathcal{P}_n$	Pauli group	X, Z

It is shown that any element of the Borel group admits representation by a canonical quantum circuit,

$$F(O,\Gamma,\Delta) = O\prod_{i=1}^{n} \mathrm{P}_i^{\Gamma_{i,i}} \prod_{1\leq i<j\leq n} \mathrm{Cz}_{i,j}^{\Gamma_{i,j}} \prod_{1\leq i<j\leq n} \mathrm{CNOT}_{i,j}^{\Delta_{j,i}}$$

where $O \in \mathcal{P}_n$ is a Pauli operator, $\Gamma, \Delta \in \mathbb{F}_2^{n\times n}$ are matrices over the binary field, Γ is symmetric, Δ is lower-triangular and unitdiagonal, i.e., $\Delta_{i,j} = 0$ for $i < j$ and $\Delta_{i,i} = 1$ for all i. Actually $F(O,\Gamma,\Delta) \in \mathcal{F}_n$, but $F(O,\Gamma,\Delta) \in \mathcal{B}_n$ iff Δ satisfys conditions we give. Then an important theorem in paper is as follows:

Theorem 3.1. *Any Clifford operator $U \in \mathcal{C}_n$ can be uniquely written as*

$$U = F(Id,\Gamma,\Delta) \cdot \left(\prod_{i=1}^{n} \mathrm{H}_i^{h_i}\right) S \cdot F\left(O',\Gamma',\Delta'\right)$$

where $h_i \in \{0,1\}, S \in \mathcal{S}_n$ is a permutation of n qubits, and $F(Id,\Gamma,\Delta), F\left(O',\Gamma',\Delta'\right)$ are elements of the Borel group $\mathcal{B}_n$ such that the matrices Γ, Δ obey the following rules for all $i, j \in [1..n]$:
C1 *if $h_i = 0$ and $h_j = 0$, then $\Gamma_{i,j} = 0$;*
C2 *if $h_i = 1$ and $h_j = 0$ and $S(i) > S(j)$, then $\Gamma_{i,j} = 0$;*
C3 *if $h_i = 0$ and $h_j = 0$ and $S(i) > S(j)$, then $\Delta_{i,j} = 0$;*
C4 *if $h_i = 1$ and $h_j = 1$ and $S(i) < S(j)$, then $\Delta_{i,j} = 0$;*
C5 *if $h_i = 1$ and $h_j = 0$, then $\Delta_{i,j} = 0$.*
The canonical form be computed in time poly(n), given the stabilizer tableaux of U.

The authors proved the existence and the uniqueness of the canonical form above, starting from the Bruhat decomposition of the Clifford group [5,section IV]. It is an improvement of that paper as well. Then they explain how to compute the canonical form which needs more detail tricks we won't explain all here. In terms of the structure of Clifford group, generation of random Clifford operator is just an application of it. Once you have a better understanding or a more concise characterization of Clifford group, you will probably come up with more ideas for sovling other problems such as how to give an efficient circuit implementation of a Clifford, too.

3.3 Generation of random Clifford operators

After getting a canonical form of a Clifford. The authors show how to solve our problem. Fist, Given a bit string $h \in \{0,1\}^n$ and a permutation $S \in \mathcal{S}_n$, define

$$W = \left(\prod_{i=1}^{n} \mathrm{H}_i^{h_i} \right) S$$

The full Clifford group $\mathcal{C}_n$ is a disjoint union of subsets $\mathcal{B}_n W \mathcal{B}_n$. We will need a normalized probability distribution

$$P_n(h,S) = \frac{|\mathcal{B}_n W \mathcal{B}_n|}{|\mathcal{C}_n|}$$

With more definitions and details, the authors find that this is a natural 'symplectic' analogue of the well-known Mallows distribution on the symmetric group $\mathcal{S}_n$ describing qubit permutations. And by generating h, S per quantum Mallows distribution $P_n(h,S)$, our random n-qubit Clifford operator picking task can be achieved. The Algorithm step by step is presented in the appendix.

4 Circuit Implemention of a Clifford Operator

In this section, we will talk about how to compose a known Clifford Operator into a circuit. Based on this two paper, similar to the task of generaing a Clifford Operator, we come up with two simple methods to turn our Clifford Operators into circuits.

4.1 Method 1

The method 1 is related to the first paper. After applying the algorithm in the first paper, we get a symplectic matrix form of a Clifford Operator. Recall that there is a significant idea of a Clifford Operator named ***tableaux*** in [2]. Recall that given a pure quantum state $|\psi\rangle$, we say a unitary matrix U stabilizes $|\psi\rangle$ if $|\psi\rangle$ is an eigenvector of U with eigenvalue 1, or equivalently if $U|\psi\rangle = |\psi\rangle$ where we do not ignore global phase. The key idea of the stabilizer formalism is to represent a quantum state $|\psi\rangle$, not by a vector of amplitudes, but by a stabilizer group, consisting of unitary matrices that stabilize $|\psi\rangle$. Remarkably, though, a large and interesting class of quantum states can be specified uniquely by much smaller stabilizer groups, specifically, the intersection of stabilzers of ψ with the Pauli group. A state is call a *stabilizer state* if it is obatinable by applying a Clifford on $|0\rangle^{\otimes n}$. And a ***tableau*** of a state consists of its *stabilizer* and *destabilizer* inside the Pauli group. For instance, a tableau for $|00\rangle$ is

$$\left(\begin{array}{cc|cc|c} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right)$$

Rows 1 to n of the tableau represent the destabilizer generators $R_1, \cdots, R_n$, and rows $n+1$ to $2n$ represent the stabilizer generators $R_{n+1}, \cdots, R_{2n}$. We can use a row to present a basis of Pauli, for instance, $(1,0,0,0,0)$ for two qubits means X_1 and $(1,1,1,1,0)$ means $X_1X_2Z_1Z_2$ where they all have phase 0. A more general tableau looks like this:

$$\left(\begin{array}{ccc|ccc|c} x_{11} & \cdots & x_{1n} & z_{11} & \cdots & z_{1n} & r_1 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{n1} & \cdots & x_{nn} & z_{n1} & \cdots & z_{nn} & r_n \\ \hline x_{(n+1)1} & \cdots & x_{(n+1)n} & z_{(n+1)1} & \cdots & z_{(n+1)n} & r_{n+1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{(2n)1} & \cdots & x_{(2n)n} & z_{(2n)1} & \cdots & z_{(2n)n} & r_{2n} \end{array}\right)$$

The tableau for $|00\rangle$ shows that the stabilizer of $|00\rangle$ is Z_1I and IZ_2. Actually it is obviously that $Z|0\rangle = |0\rangle$. So what is the tableau for ψ when we make a Clifford Operator C act on $|0\rangle$? Consider a stabilizer P of $|\psi\rangle : PC|0\rangle = C|0\rangle$, it follows $C^\dagger PC|0\rangle = |0\rangle$. Since the stabilizer of $|0\rangle$ is Z, we have $C^\dagger PC = Z$ and $P = CZC^\dagger$.

In conclusion, given a Clifford Operator C, there is a natural map between C and a stabilizer state $|\psi\rangle : |\psi\rangle = C|0\rangle^{\otimes n}$. And the tableau of $|\psi\rangle$ would be obtain easily by check the result of C's action on

$$\{X_1I\cdots I, IX_2I\cdots I, \cdots\cdots, I\cdots Z_1I\cdots I, \cdots\cdots, I\cdots\cdots Z_n\}$$

The first half of the outcome will be destabilizer of $|\psi\rangle$ and the left half will be stabilizer of $|\psi\rangle$. Since the tableau of $|00\rangle^{\otimes n}$ is Identity matrix with dimension n(ignoring the phase part), when we do the transformation to make tableau of $|\psi\rangle$ identity, this process reflects what $C^\dagger$ do. And paper [2] shows that this matrix transformation can be done step by step by clifford gate, which solves our problem of decompose C into a circuit with clifford gates.

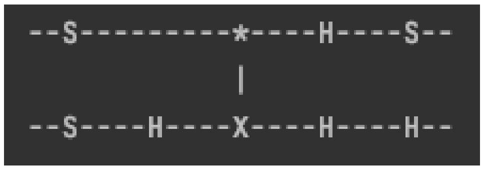
4.2 Method 2

The method 2 is related to the second paper. So as its idea of generaing random Clifford Operator, it is pretty straightforward to get the circuit of a Clifford since its canonical form reflects its gates form directly. This is an advantage we said before about this paper using a canonical form to represent a Clifford Operator rather than a map. Specifically, during our generation of a random Clifford Operator, we randomly generate $\Gamma, \Gamma', \Delta, \Delta'$ and h_i, S according to the Mallows distribution. Then we can naturally buit the circuit according to rules before:

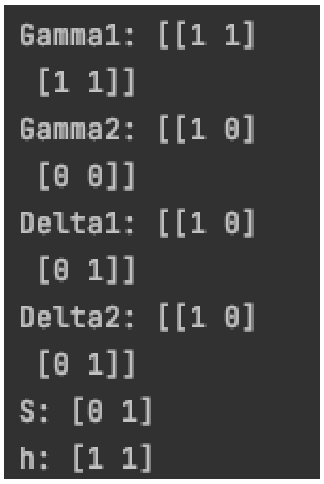
$$F(O, \Gamma, \Delta) = O \prod_{i=1}^{n} \mathrm{P}_i^{\Gamma_{i,i}} \prod_{1 \leq i < j \leq n} \mathrm{Cz}_{i,j}^{\Gamma_{i,j}} \prod_{1 \leq i < j \leq n} \mathrm{CNOT}_{i,j}^{\Delta_{j,i}}$$

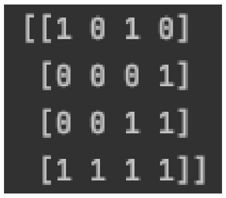
$$U = F(Id, \Gamma, \Delta) \cdot \left(\prod_{i=1}^{n} \mathrm{H}_i^{h_i} \right) S \cdot F(O', \Gamma', \Delta')$$

For example, in our experiment, we first randomly generate a Clifford Operator and use this method to get the circuit:


```
--S---------*----H----S--
            |
--S----H----X----H----H--
```

What we did is to add gate obeying the rules in equations above with $\Gamma, \Gamma', \Delta, \Delta', s, h$ as following:(the *tableau* of this Clifford Operator is shown right)


```
Gamma1: [[1 1]
 [1 1]]
Gamma2: [[1 0]
 [0 0]]
Delta1: [[1 0]
 [0 1]]
Delta2: [[1 0]
 [0 1]]
S: [0 1]
h: [1 1]
```


```
[[1 0 1 0]
 [0 0 0 1]
 [0 0 1 1]
 [1 1 1 1]]
```

4.2.1 Weakness of Method 2

One of the weakness of this method is what we are doing so far is based on that we are supposed to generate a random uniformly distributed Clifford Operator first. What if we don't have this task beforehand, and just want to decompose a given Clifford Operator? The authors talk about how to compute the canonical form of any Clifford Operator in the paper, but the complexity of that process is totally another story, which is not as easy as the situation we talk here to tell and understand.

5 Conclusion

Both this two papers give ways to gain a random uniformly distributed n-qubit Clifford operator. The fisrt one's advantage is that all a user need is to sample an integer index between 0 and $|\mathcal{C}_n| - 1$ to map into a element of $\mathcal{C}_n$. The second one is faster and more general, since the canonical form the provide can be used in other probelms related to Clifford as well. For instance, it is super convinient to decompose a Clifford Operator after we randomly generated it. And they also describe how variants of the canonical form can be used to implement arbitrary Clifford unitaries with a circuit depth of at most 9n on a linear nearest-neighbor architecture.

References

[1] Robert Koenig and John A Smolin. How to efficiently select an arbitrary Clifford group element. Journal of Mathematical Physics, 55(12):122202, 2014.

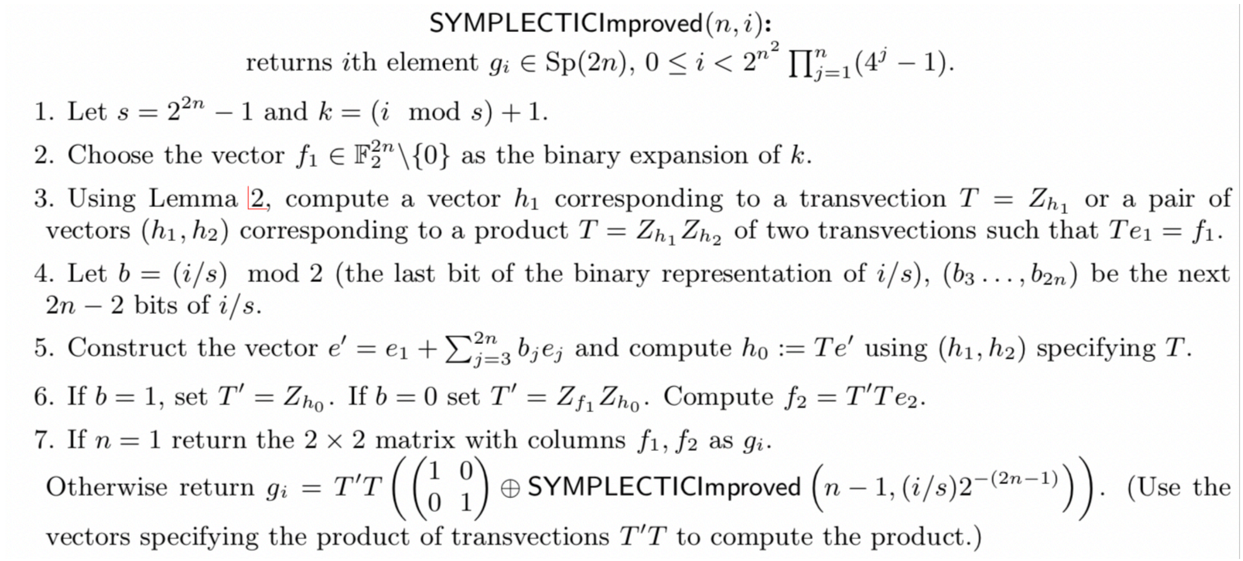
[2] Scott Aaronson and Daniel Gottesman. Improved simulation of stabilizer circuits. Physical Review A, 70(5):052328, 2004.

[3] D. DiVincenzo, D.W. Leung, and B.M. Terhal. Quantum data hiding. IEEE Trans. Inf. Th., 48(3):580–599, 2002.

[4] S. Bravyi and D. Maslov, Hadamard-free circuits expose the structure of the Clifford group. arXiv:2003.09412 [quant-ph] <https://arxiv.org/abs/2003.09412>

[5] Dmitri Maslov and Martin Roetteler. Shorter stabilizer circuits via Bruhat decomposition and quantum circuit transformations. IEEE Transactions on Information Theory, 64(7):4729–4738, 2018.

6 Appendix

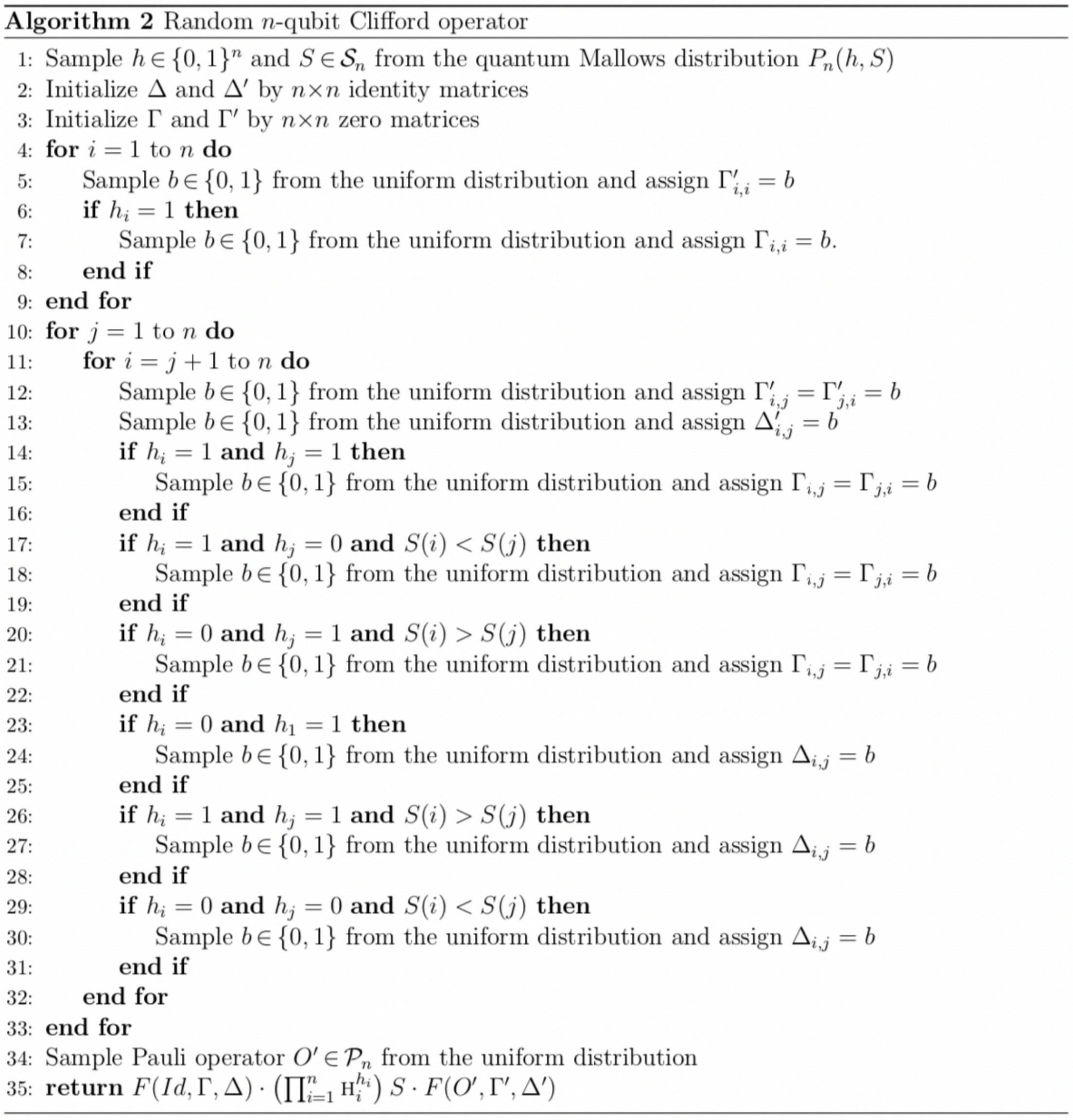


SYMPLECTICImproved(n, i):

returns ith element $g_i \in \mathrm{Sp}(2n)$, $0 \leq i < 2^{n^2} \prod_{j=1}^{n}(4^j - 1)$.

1. Let $s = 2^{2n} - 1$ and $k = (i \mod s) + 1$.
2. Choose the vector $f_1 \in \mathbb{F}_2^{2n} \backslash \{0\}$ as the binary expansion of k.
3. Using Lemma 2, compute a vector h_1 corresponding to a transvection $T = Z_{h_1}$ or a pair of vectors (h_1, h_2) corresponding to a product $T = Z_{h_1} Z_{h_2}$ of two transvections such that $Te_1 = f_1$.
4. Let $b = (i/s) \mod 2$ (the last bit of the binary representation of i/s), $(b_3 \ldots, b_{2n})$ be the next $2n - 2$ bits of i/s.
5. Construct the vector $e' = e_1 + \sum_{j=3}^{2n} b_j e_j$ and compute $h_0 := Te'$ using (h_1, h_2) specifying T.
6. If $b = 1$, set $T' = Z_{h_0}$. If $b = 0$ set $T' = Z_{f_1} Z_{h_0}$. Compute $f_2 = T'Te_2$.
7. If $n = 1$ return the 2×2 matrix with columns f_1, f_2 as g_i.
   Otherwise return $g_i = T'T\left(\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \oplus \text{SYMPLECTICImproved}\left(n-1, (i/s)2^{-(2n-1)}\right)\right)$. (Use the vectors specifying the product of transvections $T'T$ to compute the product.)

Figure 1: Algorithm in paper1



Algorithm 2 Random n-qubit Clifford operator

```
Sample $h \in \{0,1\}^n$ and $S \in \mathcal{S}_n$ from the quantum Mallows distribution $P_n(h, S)$
Initialize $\Delta$ and $\Delta'$ by $n \times n$ identity matrices
Initialize $\Gamma$ and $\Gamma'$ by $n \times n$ zero matrices
for $i = 1$ to $n$ do
    Sample $b \in \{0,1\}$ from the uniform distribution and assign $\Gamma'_{i,i} = b$
    if $h_i = 1$ then
        Sample $b \in \{0,1\}$ from the uniform distribution and assign $\Gamma_{i,i} = b$.
    end if
end for
for $j = 1$ to $n$ do
    for $i = j + 1$ to $n$ do
        Sample $b \in \{0,1\}$ from the uniform distribution and assign $\Gamma'_{i,j} = \Gamma'_{j,i} = b$
        Sample $b \in \{0,1\}$ from the uniform distribution and assign $\Delta'_{i,j} = b$
        if $h_i = 1$ and $h_j = 1$ then
            Sample $b \in \{0,1\}$ from the uniform distribution and assign $\Gamma_{i,j} = \Gamma_{j,i} = b$
        end if
        if $h_i = 1$ and $h_j = 0$ and $S(i) < S(j)$ then
            Sample $b \in \{0,1\}$ from the uniform distribution and assign $\Gamma_{i,j} = \Gamma_{j,i} = b$
        end if
        if $h_i = 0$ and $h_j = 1$ and $S(i) > S(j)$ then
            Sample $b \in \{0,1\}$ from the uniform distribution and assign $\Gamma_{i,j} = \Gamma_{j,i} = b$
        end if
        if $h_i = 0$ and $h_1 = 1$ then
            Sample $b \in \{0,1\}$ from the uniform distribution and assign $\Delta_{i,j} = b$
        end if
        if $h_i = 1$ and $h_j = 1$ and $S(i) > S(j)$ then
            Sample $b \in \{0,1\}$ from the uniform distribution and assign $\Delta_{i,j} = b$
        end if
        if $h_i = 0$ and $h_j = 0$ and $S(i) < S(j)$ then
            Sample $b \in \{0,1\}$ from the uniform distribution and assign $\Delta_{i,j} = b$
        end if
    end for
end for
Sample Pauli operator $O' \in \mathcal{P}_n$ from the uniform distribution
return $F(Id, \Gamma, \Delta) \cdot \left(\prod_{i=1}^{n} \mathrm{H}_i^{h_i}\right) S \cdot F(O', \Gamma', \Delta')$
```

Figure 2: Algorithm in paper2