npj | Quantum Information

www.nature.com/npjqi

ARTICLE OPEN

Check for updates

Optimal two-qubit circuits for universal fault-tolerant quantum computation

Andrew N. Glaudell[1,2,3,4]✉, Neil J. Ross[5] and Jacob M. Taylor[1,2]

We study two-qubit circuits over the Clifford+CS gate set, which consists of the Clifford gates together with the controlled-phase gate CS = diag(1, 1, 1, *i*). The Clifford+CS gate set is universal for quantum computation and its elements can be implemented fault-tolerantly in most error-correcting schemes through magic state distillation. Since non-Clifford gates are typically more expensive to perform in a fault-tolerant manner, it is often desirable to construct circuits that use few CS gates. In the present paper, we introduce an efficient and optimal synthesis algorithm for two-qubit Clifford+CS operators. Our algorithm inputs a Clifford+CS operator *U* and outputs a Clifford+CS circuit for *U*, which uses the least possible number of CS gates. Because the algorithm is deterministic, the circuit it associates to a Clifford+CS operator can be viewed as a normal form for that operator. We give an explicit description of these normal forms and use this description to derive a worst-case lower bound of $5\log_2(\frac{1}{\epsilon}) + O(1)$ on the number of CS gates required to ϵ-approximate elements of SU(4). Our work leverages a wide variety of mathematical tools that may find further applications in the study of fault-tolerant quantum circuits.

npj Quantum Information (2021)7:103 ; https://doi.org/10.1038/s41534-021-00424-z

INTRODUCTION

In the context of fault-tolerant quantum computing, operations from the Clifford group are relatively easy to perform and are therefore considered inexpensive. In contrast, operations that do not belong to the Clifford group are complicated to execute fault-tolerantly because they require resource-intensive distillation protocols[1]. Since non-Clifford operations are necessary for universal quantum computing, it has become standard to use the number of non-Clifford gates in a circuit as a measure of its cost. This fault-tolerant perspective on the cost of circuits has profoundly impacted the field of quantum compiling and significant efforts have been devoted to minimizing the number of non-Clifford operations in circuits.

An important problem in quantum compiling is the problem of *exact synthesis*: given an operator *U* known to be exactly representable over some gate set *G*, find a circuit for *U* over *G*. An *exact synthesis algorithm* is a constructive solution to this problem. When the gate set *G* is an extension of the Clifford group, it is desirable that the exact synthesis algorithm for *G* be efficient and produce circuits that use as few non-Clifford gates as possible.

In the past few years, methods from algebraic number theory have been successfully applied to the exact synthesis problem associated to a variety of single-qubit[2–8] and single-qutrit[9–12] gate sets. In many cases, the resulting exact synthesis algorithms efficiently produce circuits that are *optimal*, in the sense that they use the least possible number of non-Clifford gates. These powerful exact synthesis methods were central in the development of good unitary approximation methods, which play a key role in the compilation of practical quantum programs[2,3,7,8,13,14].

Exact synthesis algorithms also exist for various instantiations of the multi-qubit compiling problem, though each suffers shortcomings in some respect. Optimal algorithms for two-qubit circuits over continuous gate sets have been known for a number of years[15,16]. Unfortunately, such gate sets are not well-suited for fault-tolerant quantum computing. Multi-qubit exact synthesis algorithms for universal and fault-tolerant gate sets were introduced more recently[17–27]. Some of these algorithms, such as[17,19–21,24–26], are proper synthesis algorithms: they input a unitary matrix and produce a circuit. Some other of these algorithms, such as[18,22,23,27], are better referred to as *re*-synthesis algorithms: they input a circuit and produce an optimized circuit. Of course, a re-synthesis algorithm can be used in conjunction with a synthesis algorithm to obtain an alternative (and typically better) synthesis algorithm. While the algorithms of refs. [17,20] are far from optimal, the algorithms of[19,21,24,26] synthesize provably optimal circuits by cleverly utilizing certain properties of fault-tolerant gate sets containing the Clifford group. However, the runtimes of these optimal synthesis algorithms are exponential in both qubit count and optimal circuit length. Powerful heuristics were introduced in[26] achieving polynomial scaling with optimal circuit length. Unfortunately, even this improved heuristic algorithm takes thousands of seconds to compute optimal two-qubit circuits of practical size (40 non-Clifford operations) on modest hardware.

Not only are these multi-qubit exact synthesis algorithms impractical in many cases, they also fail to shed much light on the *structure* of optimal circuits. In the single-qubit case, intimate knowledge of this structure for certain gate sets was developed by describing optimal circuits via regular expressions or, equivalently, automata[28]. Such descriptions are of theoretical interest, but also have practical consequences. In particular, for certain single-qubit gate sets these descriptions allowed researchers to derive a rigorous lower-bound on the number of non-Clifford gates required to approximate typical elements of SU(2)[29]. Analogous statements about approximations of multi-qubit unitaries have eluded researchers thus far.

In the present paper, we introduce an efficient and optimal exact synthesis algorithm for a two-qubit gate set that is appropriate for universal and fault-tolerant quantum computing.

[1]Institute for Advanced Computer Studies and Joint Center for Quantum Information and Computer Science, University of Maryland, College Park, MD, USA. [2]Joint Quantum Institute, University of Maryland, College Park, MD, USA. [3]Booz Allen Hamilton, Annapolis Junction, MD, USA. [4]Department of Mathematical Sciences, George Mason University, Fairfax, VA, USA. [5]Department of Mathematics and Statistics, Dalhousie University, Halifax, NS, Canada. ✉email: Glaudell_Andrew@bah.com

We focus on two-qubit circuits over the Clifford+CS gate set, which consists of the Clifford gates together with the non-Clifford controlled-phase gate $\mathrm{CS} = \mathrm{diag}(1, 1, 1, i)$. The CS gate has received recent attention as an alternative to the *T*-gate in methods for fault-tolerant quantum computing[30,31] and due to its natural implementation as an entangling operation in certain superconducting qubit systems[32–35] whose fidelity is approaching that of single-qubit gates[36,37]. Our algorithm produces an optimal circuit in a number of arithmetic operations linear in the length of the optimal decomposition. This is unlike existing multi-qubit synthesis methods. Moreover, because our algorithm is deterministic, the circuit it associates to a Clifford+CS operator can be viewed as a normal form for that operator. We give an explicit description of these normal forms in the language of automata and use this description to derive a worst-case lower bound of $5\log_2(\frac{1}{\epsilon}) + O(1)$ on the number of CS gates required to ϵ-approximate elements of SU(4). A Mathematica package implementing our algorithm is freely available online[38]. This code is very efficient, synthesizing optimal circuits of CS-count 10000 in 1.2 ± 0.1 s on modest hardware.

The paper is structured as follows. We first introduce a convenient set of generators in Section "Generators". Then, in Section "The isomorphism SU(4) ≅ Spin(6)", we describe the exceptional isomorphism SU(4) ≅ Spin(6). In Section "Exact synthesis", we leverage this isomorphism to introduce an exact synthesis algorithm for Clifford+CS operators. In Sections "Automata as tools for describing normal forms" and "The structure of normal forms", we use the theory of automata to study the structure of the circuits produced by the exact synthesis algorithm. We take advantage of this structure in Section "Lower bounds" to establish a worst-case lower bound on the number of non-Clifford resources required to ϵ-approximate elements of SU(4) using Clifford+CS circuits. Finally, we conclude and discuss avenues for future work in Section "Discussion".

RESULTS

Generators

Throughout, we use $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{R}$, and $\mathbb{C}$ to denote the usual collection of numbers, $\mathbb{Z}_p$ to denote the collection integers modulo p, and $\mathbb{Z}[i]$ to denote the collection of Gaussian integers (the complex numbers with integer real and imaginary parts). We write ρ for the canonical homomorphism $\mathbb{Z} \to \mathbb{Z}_2$ (if $n \in \mathbb{Z}$ then $\rho(n)$ is the parity of n). For two integers $n \le m$, we write $[n, m]$ for the set $\{n, \ldots, m\} \subseteq \mathbb{Z}$ and simply write $[m]$ for $[1, m]$. We view scalars and vectors as matrices so that any concept defined for matrices of arbitrary dimensions also applies to scalars and vectors. Finally, for readability, we use the symbol $\cdot$ to denote the zero entries of a matrix.

The single-qubit *Pauli* gates *X*, *Y*, and *Z* are defined as

$$X = \begin{bmatrix} \cdot & 1 \\ 1 & \cdot \end{bmatrix}, \quad Y = \begin{bmatrix} \cdot & -i \\ i & \cdot \end{bmatrix}, \quad \text{and} \quad \mathrm{Z} = \begin{bmatrix} 1 & \cdot \\ \cdot & -1 \end{bmatrix}.$$

These gates generate the *single-qubit Pauli group* $\{i^a P\,;\, a \in \mathbb{Z}_4 \text{ and } \mathrm{P} \in \{\mathrm{I}, \mathrm{X}, \mathrm{Y}, \mathrm{Z}\}\}$. The *two-qubit Pauli group*, which we denote by $\mathcal{P}$, is defined as $\mathcal{P} = \{i^a (P \otimes Q)\,;\, a \in \mathbb{Z}_4 \text{ and } P, Q \in \{I, X, Y, Z\}\}$. The *Clifford* gates *H*, *S*, and CZ are defined as

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \; S = \begin{bmatrix} 1 & \cdot \\ \cdot & i \end{bmatrix}, \quad \text{and} \quad \mathrm{CZ} = \begin{bmatrix} 1 & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot \\ \cdot & \cdot & \cdot & -1 \end{bmatrix}.$$

These gates are known as the *Hadamard* gate, the *phase* gate, and the *controlled-Z* gate, respectively. The *single-qubit Clifford group* is generated by *H* and *S* and contains the primitive 8th root of unity $\omega = e^{\frac{i\pi}{4}}$. The *two-qubit Clifford group*, which we denote by $\mathcal{C}$, consists of the operators which can be represented by a two-qubit circuit over the gate set $\{H, S, \mathrm{CZ}\}$. Equivalently, $\mathcal{C}$ is generated by $H \otimes I$, $I \otimes H$, $S \otimes I$, $I \otimes S$, and CZ. Up to global phases, the Clifford groups are the normalizers of the Pauli groups.

Clifford gates are well-suited for fault-tolerant quantum computation but the Clifford group is not universal. One can obtain a universal group by extending $\mathcal{C}$ with the *controlled-phase gate* CS defined as

$$\mathrm{CS} = \begin{bmatrix} 1 & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot \\ \cdot & \cdot & \cdot & i \end{bmatrix}.$$

In what follows, we focus on the group $\mathcal{G}$ of operators which can be represented by a two-qubit circuit over the universal gate set $\{H, S, \mathrm{CZ}, \mathrm{CS}\}$. Equivalently, $\mathcal{G}$ is the group generated by $H \otimes I$, $I \otimes H$, $S \otimes I$, $I \otimes S$, CZ, and CS. We have $\mathcal{P} \subseteq \mathcal{C} \subseteq \mathcal{G}$. We sometimes refer to $\mathcal{G}$ as the *Clifford + CS* group or *Clifford + controlled-phase* group. We know from[17] that $\mathcal{G}$ is the group of 4×4 unitary matrices of the form

$$\frac{1}{\sqrt{2}^k} M \tag{1}$$

where $k \in \mathbb{N}$ and the entries of *M* belong to $\mathbb{Z}[i]$. In the fault-tolerant setting, the CS gate is considered vastly more expensive than any of the Clifford gates. As a result, the cost of a Clifford + CS circuit is determined by its *CS-count*: the number of CS gates that appear in the circuit. Our goal is to find circuits for the elements of $\mathcal{G}$ that are optimal in CS-count.

We start by introducing a generalization of the CS gate which will be helpful in describing the elements of $\mathcal{G}$.

Definition 2.1
Let *P* and *Q* be distinct elements of $\mathcal{P}\backslash\{I\}$ such that *P* and *Q* are Hermitian and $PQ = QP$. Then $R(P, Q)$ is defined as

$$R(P, Q) = \exp\left(\frac{i\pi}{2}\left(\frac{I - P}{2}\right)\left(\frac{I - Q}{2}\right)\right).$$

We have $R(Z \otimes I, I \otimes Z) = \mathrm{CS}$. Moreover, since $\mathcal{C}$ normalizes $\mathcal{P}$ and $CR(P, Q)C^\dagger = R(CPC^\dagger, CQC^\dagger)$ for every $C \in \mathcal{C}$, we know that $R(P, Q) \in \mathcal{G}$ for every appropriate $P, Q \in \mathcal{P}$. We record some important properties of the $R(P, Q)$ gates in the lemma below. Because the proof of the lemma is tedious but relatively straightforward, it is given in Supplementary Note 2.

Lemma 2.2
Let $C \in \mathcal{C}$ and let *P*, *Q*, and *L* be distinct elements of $\mathcal{P} \setminus \{I\}$. Assume that *P*, *Q*, and *L* are Hermitian and that $PQ = QP$, $PL = LP$, and $QL = -LQ$. Then the following relations hold:

$$CR(P, Q)C^\dagger = R(CPC^\dagger, CQC^\dagger), \tag{2}$$

$$R(P, Q) = R(Q, P), \tag{3}$$

$$R(P, -PQ) = R(P, Q), \tag{4}$$

$$R(P, -Q) \in R(P, Q)\mathcal{C}, \tag{5}$$

$$R(P, Q)^2 \in \mathcal{C}, \text{ and} \tag{6}$$

$$R(P, L)R(P, Q) = R(P, Q)R(P, iQL). \tag{7}$$

We will use the $R(P, Q)$ gates of Definition 2.1 to define normal forms for the elements of $\mathcal{G}$. The equivalences given by Lemma

$$\begin{array}{lllll} R(X\otimes I, I\otimes X) & R(Y\otimes I, I\otimes Y) & R(Z\otimes I, I\otimes Z) & R(Y\otimes I, I\otimes Z) & R(Z\otimes I, I\otimes Y) \\ R(Z\otimes I, I\otimes X) & R(X\otimes I, I\otimes Z) & R(X\otimes I, I\otimes Y) & R(Y\otimes I, I\otimes X) & R(X\otimes X, Y\otimes Y) \\ R(X\otimes X, Z\otimes Y) & R(Z\otimes X, Y\otimes Y) & R(Y\otimes X, X\otimes Y) & R(Z\otimes X, X\otimes Y) & R(Y\otimes X, Z\otimes Y) \end{array}$$

Fig. 1 The 15 elements of $\mathcal{S}$. These operators are one suitable choice for 15 $\mathcal{R}(\mathcal{P},\mathcal{Q})$ gates which are not equivalent to each other up to right-multiplication by Cliffords. All other choices are equivalent to this one up to right-multiplication by Cliffords. The ordering of $\mathcal{S}$ is given by reading left-to-right and row-by-row.

2.2 show that it is not necessary to use every $R(P,Q)$ gate and the following definition specifies the ones we will be using.

Definition 2.3
Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be the subsets of $\mathcal{P}\times\mathcal{P}$ given below.

$$\mathcal{T}_1 = \{(P,Q)\,;\, P\in\{X\otimes I, Y\otimes I, Z\otimes I\}, Q\in\{I\otimes X, I\otimes Y, I\otimes Z\}\}$$
$$\mathcal{T}_2 = \{(P,Q)\,;\, P\in\{X\otimes X, Z\otimes X, Y\otimes X\}, Q\in\{Y\otimes Y, Z\otimes Y, X\otimes Y\}, \text{and } PQ = QP\}.$$

The set $\mathcal{S}$ is defined as $\mathcal{S} = \{R(P,Q)\,;\,(P,Q)\in\mathcal{T}_1 \text{ or } (P,Q)\in\mathcal{T}_2\}$. The set $\mathcal{S}$ contains 15 elements which are explicitly listed in Fig. 1. It can be verified that all of the elements of $\mathcal{S}$ are distinct, even up to right-multiplication by a Clifford gate. It will be helpful to consider the set $\mathcal{S}$ ordered as in Fig. 1, which is to be read left-to-right and row-by-row. We then write $\mathcal{S}_j$ to refer to the j-th element of $\mathcal{S}$. For example, $\mathcal{S}_1$ is in the top left of Fig. 1, $\mathcal{S}_5$ is in the top right, and $\mathcal{S}_{15}$ is in the bottom right. The position of $R(P,Q)$ in this ordering roughly expresses the complexity of the Clifford circuit required to conjugate CS to $R(P,Q)$.
We close this section by showing that every element of $\mathcal{G}$ can be expressed as a sequence of elements of $\mathcal{S}$ followed by a single element of $\mathcal{C}$.

Lemma 2.4
Let P and Q be distinct elements of $\mathcal{P}\setminus\{I\}$ such that P and Q are Hermitian and $PQ = QP$. Then there exists $P', Q'\in\mathcal{P}$ and $C\in\mathcal{C}$ such that $R(P',Q')\in\mathcal{S}$ and $R(P,Q) = R(P',Q')C$.
Proof
Let $P = i^p(P_1\otimes P_2)$ and $Q = i^q(Q_1\otimes Q_2)$ with $P_1, P_2, Q_1, Q_2 \in \{I, X, Y, Z\}$. Since P and Q are Hermitian, p and q must be even. Moreover, by Eqs. (3) and (5) of Lemma 2.2, we can assume without loss of generality that $p = q = 0$ so that $P = P_1\otimes P_2$ and $Q = Q_1\otimes Q_2$. Now, if one of P_1, P_2, Q_1, or Q_2 is I, then we can use Eqs. (3), (4) and (5) of Lemma 2.2 to rewrite $R(P,Q)$ as with $C\in\mathcal{C}$ and $(P',Q')\in\mathcal{T}_1$ as in Definition 2.3. If, instead, none of P_1, P_2, Q_1, or Q_2 are I, then we can reason similarly to rewrite $R(P,Q)$ as $R(P',Q')C$ with $C\in\mathcal{C}$ and $(P',Q')\in\mathcal{T}_2$. □

Proposition 2.5
Let $V\in\mathcal{G}$. Then $V = R_1\cdots R_n C$ where $C\in\mathcal{C}$ and $R_j\in\mathcal{S}$ for $j\in[n]$.
Proof
Let $V\in\mathcal{G}$. Then V can be written as $V = C_1\cdot\mathrm{CS}\cdot C_2\cdot\mathrm{CS}\cdot\ldots\cdot C_n\cdot\mathrm{CS}\cdot C_{n+1}$ where $C_j\in\mathcal{C}$ for $j\in[n+1]$. Since $\mathrm{CS} = R(Z\otimes I, I\otimes Z)$ we have

$$V = C_1\cdot R(Z\otimes I, I\otimes Z)\cdot C_2\cdot R(Z\otimes I, I\otimes Z)\cdot\ldots\cdot C_n \cdot R(Z\otimes I, I\otimes Z)\cdot C_{n+1}. \tag{8}$$

Now, by Eq. (2) of Lemma 2.2, $C_1R(Z\otimes I, I\otimes Z) = C_1R(Z\otimes I, I\otimes Z)C_1^\dagger C_1 = R(P,Q)C_1$ for some $P, Q\in\mathcal{P}$. We can then apply Lemma 2.4 to get

$$C_1R(Z\otimes I, I\otimes Z) = R(P,Q)C_1 = R(P',Q')CC_1 = R(P',Q')C'$$

with $C' = CC_1\in\mathcal{C}$ and $R(P',Q')\in\mathcal{S}$. Hence, setting $R_1 = R(P',Q')$ and $C' = C'C_2$, Eq. (8) becomes

$$V = R_1\cdot C'\cdot R(Z\otimes I, I\otimes Z)\cdot\ldots\cdot C_n\cdot R(Z\otimes I, I\otimes Z)\cdot C_{n+1}$$

and we can proceed recursively to complete the proof.

The Isomorphism SU(4) ≅ Spin(6)

In this section, we describe the exceptional isomorphism SU(4) ≅ Spin(6) which will allow us to rewrite two-qubit operators as elements of SO(6). Consider some element U of SU(4). Then U acts on $\mathbb{C}^4$ by left-multiplication. Moreover, this action is norm-preserving. Now let $\{e_j\}$ be the standard orthonormal basis of $\mathbb{C}^4$. From this basis, we construct an alternative six-component basis using the *wedge product*.

Definition 2.6
(Wedge product). Let $a\wedge b$ be defined as the *wedge product* of a and b. Wedge products have the following properties given vectors $a, b, c\in\mathbb{C}^n$ and $\alpha,\beta\in\mathbb{C}$:

- Anticommutativity: $a\wedge b = -b\wedge a$.
- Associativity: $(a\wedge b)\wedge c = a\wedge(b\wedge c)$.
- Bilinearity: $(\alpha a+\beta b)\wedge c = \alpha(a\wedge c)+\beta(b\wedge c)$.

Note that the anticommutation of wedge products implies that $a\wedge a = 0$. We say that $v_1\wedge\cdots\wedge v_k\in\bigwedge^k\mathbb{C}^n$ for $v_j\in\mathbb{C}^n$. To compute the inner product of two wedge products $v_1\wedge\cdots\wedge v_k$ and $w_1\wedge\cdots\wedge w_k$, we compute

$$\langle v_1\wedge\cdots\wedge v_k, w_1\wedge\cdots\wedge w_k\rangle = \det\big(\langle v_q, w_r\rangle\big)$$

where $\langle v_q, w_r\rangle$ is the entry in the q-th row and r-th column of a $k\times k$ matrix.
Remark 2.7
The magnitude of a wedge product of n vectors can be thought of as the n dimensional volume of the parallelotope constructed from those vectors. The orientation of the wedge product defines the direction of circulation around that parallelotope by those vectors. The wedge product of two vectors in $\mathbb{C}^4$ can be decomposed into a six-component basis as anticommutativity reduces the 16 potential wedge products of elements of $\{e_j\}$ to six. We choose this basis as

$$B = \{s_{-,12,34}, s_{+,12,34}, s_{-,23,14}, s_{+,24,13}, s_{-,24,13}, s_{+,23,14}\} \tag{9}$$

where

$$s_{\pm,ij,kl} = \frac{I^{\frac{1\mp1}{2}}}{\sqrt{2}}\left(e_i\wedge e_j \pm e_k\wedge e_l\right). \tag{10}$$

We note that B is an orthonormal basis and we assume that B is ordered as in Eq. (9).

Definition 2.8
Let $U\in$ SU(4) and $\overline{U}$ be its representation in the transformed basis. Let $v, w\in\mathbb{C}^4$ with $v\wedge w\in\bigwedge^2\mathbb{C}^4$. Then the actions of U and $\overline{U}$ are related by

$$\overline{U}(v\wedge w) = (Uv)\wedge(Uw).$$

To avoid confusion, we use an overline, as in $\overline{O}$, to denote the SO(6) representation of an operator or set of operators O. We are now equipped to define the transformation from SU(4) to SO(6).

Definition 2.9
Let $U\in$ SU(4) and let $j, k\in[6]$. Then the entry in the j-th row and k-th column of the SO(6) representation $\overline{U}$ of U is

$$\overline{U}_{j,k} = \langle B_j, \overline{U}B_k\rangle \tag{11}$$

where B_j is the jth element in the ordered basis B, the action of $\overline{U}$ on B_k is defined by Definitions 2.6 and 2.8, and the inner product is defined by Definitions 2.6.

As an illustration of the process specified in Definition 2.9, we explicitly calculate the SO(6) representation of a Clifford+CS operator in Supplementary Note 1. Moreover, we provide code to compute this isomorphism for any input with our Mathematica package[38].

Remark 2.10

The fact that this isomorphism yields special orthogonal operators is ultimately due to the fact that the Dynkin diagrams for the Lie algebras of SU(4), Spin(6), and SO(6) are equivalent. However, this fact can be easily illustrated through the Euler decomposition of SU(4)[39]. Direct calculation of $\overline{U}$ for the operator

$$U = \begin{bmatrix} 1 & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & a & \cdot \\ \cdot & \cdot & \cdot & a^* \end{bmatrix}$$

for $|a| = 1$ and $a = r + ic$ with $r, c \in \mathbb{R}$ yields

$$\overline{U} = \begin{bmatrix} 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & r & \cdot & \cdot & c \\ \cdot & \cdot & \cdot & r & c & \cdot \\ \cdot & \cdot & \cdot & -c & r & \cdot \\ \cdot & \cdot & -c & \cdot & \cdot & r \end{bmatrix}$$

which is explicitly in SO(6). Computation of the other 14 Euler angle rotations required for an SU(4) parameterization yields similar matrices, likewise in SO(6). Since SO(6) is a group under multiplication, the isomorphism applied to any $U \in \mathrm{SU}(4)$ yields $\overline{U} \in \mathrm{SO}(6)$.

We close this section by explicitly calculating the SO(6) representation of each of the generators of $\mathcal{G}$. We multiply the generators by overall phase factors to ensure that each operator has determinant one, and furthermore that single-qubit operators have determinant one on their single-qubit subspace. Later, when referring to gates or their SO(6) representation, we omit overall phases for readability.

Proposition 2.11

The image of the generators of $\mathcal{C}$ in SO(6) are

$$\overline{(\omega^\dagger S) \otimes I} = \begin{bmatrix} \cdot & -1 & \cdot & \cdot & \cdot & \cdot \\ 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & 1 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & 1 \end{bmatrix}, \quad \overline{I \otimes (\omega^\dagger S)} = \begin{bmatrix} 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & -1 & \cdot \\ \cdot & \cdot & \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & 1 \end{bmatrix},$$

$$\overline{(iH) \otimes I} = \begin{bmatrix} \cdot & \cdot & 1 & \cdot & \cdot & \cdot \\ \cdot & -1 & \cdot & \cdot & \cdot & \cdot \\ 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & 1 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & 1 \end{bmatrix}, \quad \overline{I \otimes (iH)} = \begin{bmatrix} 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & 1 \\ \cdot & \cdot & \cdot & \cdot & -1 & \cdot \\ \cdot & \cdot & \cdot & 1 & \cdot & \cdot \end{bmatrix},$$

$$\overline{\omega^\dagger \mathrm{CZ}} = \begin{bmatrix} \cdot & -1 & \cdot & \cdot & \cdot & \cdot \\ 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & -1 \\ \cdot & \cdot & \cdot & \cdot & -1 & \cdot \\ \cdot & \cdot & \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot & \cdot & \cdot \end{bmatrix}.$$

Proposition 2.12

The elements of $\overline{\mathcal{S}}$ are given in Fig. 2.

Exact synthesis

In this section, we leverage the isomorphism $\mathrm{SU}(4) \cong \mathrm{Spin}(6)$ described in the previous section to find optimal decompositions for the elements of $\mathcal{G}$. We will be working extensively with the matrix group

$$\mathcal{H} = \left\{ \frac{1}{\sqrt{2}^k} M \in \mathrm{SO}(6) \,;\, k \in \mathbb{N}, M \in \mathbb{Z}^{6\times 6} \right\}. \tag{12}$$

Note that $\mathcal{H} \subseteq \mathrm{SO}(6)$. Our interest in $\mathcal{H}$ stems from the following observation.

Proposition 2.13

We have $\overline{\mathcal{G}} \subseteq \mathcal{H}$.

Proof

The property holds for the generators of $\overline{\mathcal{G}}$ by Propositions 2.11 and 2.12. □

In the remainder of this section, we prove the converse of Proposition 2.13 by defining an algorithm which inputs an element of $\mathcal{H}$ and outputs a product of generators. We start by introducing a few notions that are useful in discussing the elements of $\mathcal{H}$.

Definition 2.14

Let $V \in \mathcal{H}$. We say that $\ell \in \mathbb{N}$ is a *denominator exponent* of V if $\sqrt{2}^\ell V \in \mathbb{Z}^{6\times 6}$. The least such ℓ is the *least denominator exponent* of V, which we denote by $\mathrm{lde}(V)$.

Lemma 2.15

Let $U \in \mathcal{G}$ and suppose that $\mathrm{lde}(\overline{U}) = k$. Then any Clifford+CS circuit for U has CS-count at least k.

Proof

The only generators with a factor of $1/\sqrt{2}$ in their SO(6) representation are the elements of $\mathcal{S}$. Thus, for a least denominator exponent of k there must be at least k of these operators, each of which requires a single CS gate. □

Definition 2.16

Let $V \in \mathcal{H}$ and let ℓ be a denominator exponent of V. The *ℓ-residue* of V is the binary matrix $\rho_\ell(V) \in \mathbb{Z}_2^{6\times 6}$ defined by

$$(\rho_\ell(V))_{ij} = \rho((\sqrt{2}^\ell V)_{ij})$$

where $\rho : \mathbb{Z} \to \mathbb{Z}_2$ is the canonical (parity) homomorphism.

The residue matrices introduced in Definition 2.16 are important in the definition of the exact synthesis algorithm. Indeed, the ℓ-residue of a Clifford+CS operator U determines the element of $\mathcal{S}$ to use in order to reduce the least denominator exponent of U (although not uniquely, as we discuss below). Similar residue matrices are used in the study of other fault-tolerant circuits[17,28]. Recall that if A is a set, then a *partition* of A is a collection of disjoint nonempty subsets of A whose union is equal to A. The set of all partitions of a set A is denoted $\mathscr{B}_A$. Let p and p' be two partitions of A. If every element of p is a subset of an element of p' then we say that p' is *coarser* than p and that p is *finer* than p'.

Definition 2.17

Let $N \in \mathbb{Z}_2^{6\times 6}$ be a binary matrix with rows $r_1, \ldots, r_6$ and let $p = \{p_1, \ldots, p_q\}$ be a partition of the set [6]. Then N has the *pattern* p if for any p_j in p and any $j_1, j_2 \in p_j$ we have $r_{j_1} = r_{j_2}$. In this case we also say that N has a $|p_1| \times \ldots \times |p_q|$ *pattern*.

Definition 2.18

Let $V \in \mathcal{H}$ with $\mathrm{lde}(V) = \ell$. We define the pattern map $\mathfrak{p} : \mathcal{H} \to \mathscr{B}_{[6]}$ as the function which maps V to the pattern of $\rho_\ell(V)$. We say

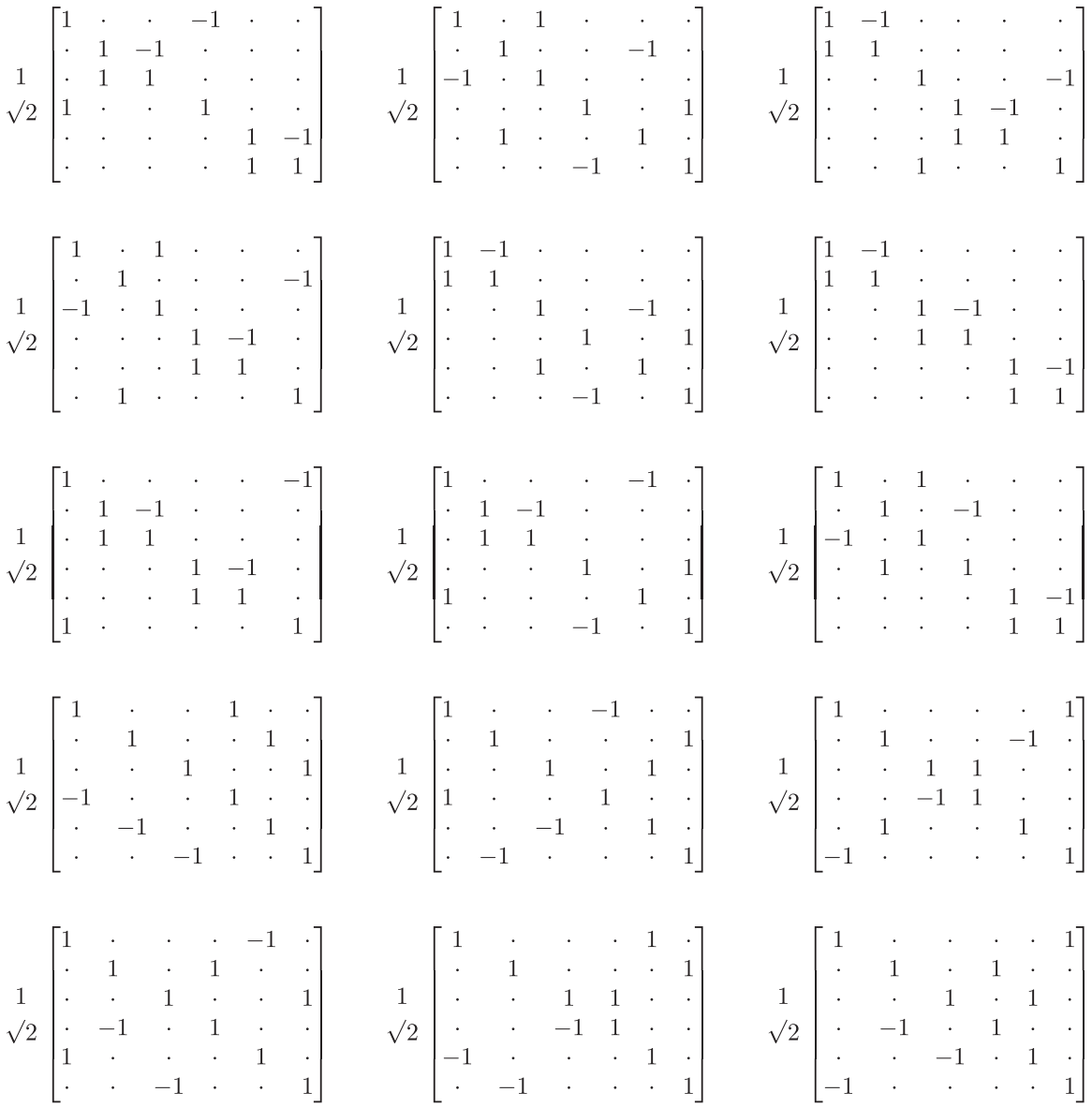


Fig. 2 The 15 elements of $\overline{\mathcal{S}}$. These are the images of the 15 elements of $\mathcal{S}$ under the action of the SU(4) ≅ Spin(6) isomorphism. Each matrix may be associated with a unique pairing of rows and columns.

that $p = \mathfrak{p}(V)$ is the pattern of V. If V_1 and V_2 are two elements of $\mathcal{H}$, we say that V_1 is *finer* than V_2 or that V_2 is *coarser* than V_1 if these statements hold for $\mathfrak{p}(V_1)$ and $\mathfrak{p}(V_2)$.

Remark 2.19
In a slight abuse of notation, we extend the pattern map to any valid representation of a Clifford+CS operator. Given a Clifford +CS operator with SU(4) representation U which can be written as a word W over the generators and with SO(6) representation $\overline{U}$, we set $\mathfrak{p}(U) = \mathfrak{p}(W) = \mathfrak{p}(\overline{U})$. This extension is unambiguous after fixing our transformation from SU(4) to SO(6), as $\mathfrak{p}$ is insensitive to relative phase changes in U. We incorporate all relational notions described in Definition 2.18 in this extension.

We now analyze the image in SO(6) of certain subsets of $\mathcal{G}$. We start by showing that the image of the Clifford group $\mathcal{C}$ is exactly the collection of elements of $\mathcal{H}$ with least denominator 0. In other words, $\overline{\mathcal{C}}$ is the group of 6-dimensional signed permutation matrices.

Lemma 2.20
Let $V \in \mathcal{H}$. Then $\mathrm{lde}(V) = 0$ if and only if $V \in \overline{\mathcal{C}}$.

Proof
The least denominator exponent of $\overline{H \otimes I}$, $\overline{I \otimes H}$, $\overline{S \otimes I}$, $\overline{I \otimes S}$, and $\overline{CZ}$ is 0. Thus, if $U \in \mathcal{C}$ then $\mathrm{lde}(\overline{U}) = 0$. For the converse, let C_1 and C_2 be the Clifford operators $(\omega^\dagger S) \otimes I$ and $(H \otimes H)(\omega^\dagger \mathrm{CZ})(Z \otimes Z)$, respectively. Then

$$\overline{C_1} = \begin{bmatrix} \cdot & -1 & \cdot & \cdot & \cdot & \cdot \\ 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & 1 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & 1 \end{bmatrix} \text{ and } \overline{C_2} = \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & -1 \\ 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & 1 & \cdot \end{bmatrix}.$$

The operators $\overline{C_1}$ and $\overline{C_2}$ generate $\{V \in \mathcal{H}\,;\, \mathrm{lde}\,(V) = 0\}$. Hence, if $V \in \mathcal{H}$ and $\mathrm{lde}(V) = 0$ then V can be expressed as a product of the image of Clifford gates. □

Lemma 2.21
Let $V \in \mathcal{H}$. Then $\mathrm{lde}(V) = 1$ if and only if $V = \overline{RC}$ for some $R \in \mathcal{S}$ and some $C \in \mathcal{C}$. Furthermore, V has a $2 \times 2 \times 2$ pattern.

Proof
The rows of V have unit norm and are pairwise orthogonal. Hence, up to a signed permutation of rows and columns, there is only one such matrix, e.g.,

$$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -1 & \cdot & \cdot & \cdot & \cdot \\ 1 & 1 & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & 1 & -1 & \cdot & \cdot \\ \cdot & \cdot & 1 & 1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & 1 & -1 \\ \cdot & \cdot & \cdot & \cdot & 1 & 1 \end{bmatrix} = \overline{\mathcal{S}}_6. \tag{13}$$

By Proposition 2.5 the proof is complete, since Clifford operators correspond to signed permutations by Lemma 2.20. □

Lemma 2.22
Let $V \in \mathcal{H}$ with $\mathrm{lde}(V) = k \geq 2$. Then V has either a $2\times2\times2$ or 2×4 pattern.
Proof
Let $V \in \mathcal{H}$. Since V is orthogonal, we have $V^\dagger V = I$. Hence, $(\sqrt{2}^k V)^\dagger (\sqrt{2}^k V) = 2^k I$. Since $k \geq 2$, this implies that the inner product of any column of $\sqrt{2}^k V$ with itself is congruent to 0 modulo 4. Similarly, the inner product of two distinct columns $\sqrt{2}^k V$ is congruent to 0 modulo 4. Letting, $M = \rho_k(V)$, we then have the column relations

$$\sum_l M_{lm}^2 = 0 \bmod 4 \tag{14}$$

$$\sum_l M_{lm} M_{ln} = 0 \bmod 2 \text{ for } m \neq n \tag{15}$$

as well as analogous row relations. For $x \in \mathbb{Z}$, $x^2 = 0 \bmod 4$ if and only if $x = 0 \bmod 2$. Hence, there must be exactly zero or four odd entries in every column (or row) of M by Eq. (14). By Eq. (15), we see that the inner product of any two distinct rows must be even. Up to a permutation of rows and columns, we can then deduce that M is one of the two matrices below, which completes the proof.

$$\begin{bmatrix} 1 & 1 & 1 & 1 & \cdot & \cdot \\ 1 & 1 & 1 & 1 & \cdot & \cdot \\ 1 & 1 & 1 & 1 & \cdot & \cdot \\ 1 & 1 & 1 & 1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \text{ or } \begin{bmatrix} 1 & 1 & 1 & 1 & \cdot & \cdot \\ 1 & 1 & 1 & 1 & \cdot & \cdot \\ 1 & 1 & \cdot & \cdot & 1 & 1 \\ 1 & 1 & \cdot & \cdot & 1 & 1 \\ \cdot & \cdot & 1 & 1 & 1 & 1 \\ \cdot & \cdot & 1 & 1 & 1 & 1 \end{bmatrix} \tag{16}$$

□

Corollary 2.23
Let $V \in \mathcal{H}$ with $\mathrm{lde}(V) = k \geq 1$. Then V has either a $2\times2\times2$ or 2×4 pattern.

Lemma 2.24
Let $V \in \mathcal{H}$ and assume that $\mathrm{lde}(V) = k \geq 1$. If $\overline{R} \in \overline{\mathcal{S}}$ is finer than V, then $\mathrm{lde}(\overline{R}^\dagger V) = k - 1$.
Proof
For simplicity, we assume that $\mathfrak{p}(\overline{R}) = \{\{1,2\},\{3,4\},\{5,6\}\}$. The cases in which $\mathfrak{p}(\overline{R})$ is another pattern are treated similarly. For $j \in [6]$, let r_j denote the rows of $\sqrt{2}^k V$. Since $\mathfrak{p}(V)$ is coarser than $\mathfrak{p}(\overline{R})$, we have $r_1 \equiv r_2$, $r_3 \equiv r_4$, $r_5 \equiv r_6$ modulo 2. This implies that $r_1 \pm r_2 \equiv r_3 \pm r_4 \equiv r_5 \pm r_6 \equiv 0$ modulo 2. Hence

$$\overline{R}^\dagger V = \frac{1}{\sqrt{2}^{k+1}}\begin{bmatrix} 1 & 1 & \cdot & \cdot & \cdot & \cdot \\ -1 & 1 & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & 1 & 1 & \cdot & \cdot \\ \cdot & \cdot & -1 & 1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & 1 & 1 \\ \cdot & \cdot & \cdot & \cdot & -1 & 1 \end{bmatrix}\begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \end{bmatrix} = \frac{1}{\sqrt{2}^{k+1}}\begin{bmatrix} r_1 - r_2 \\ r_1 + r_2 \\ r_3 - r_4 \\ r_3 + r_4 \\ r_5 - r_6 \\ r_5 + r_6 \end{bmatrix} = \frac{1}{\sqrt{2}^{k-1}}\begin{bmatrix} r_1' \\ r_2' \\ r_3' \\ r_4' \\ r_5' \\ r_6' \end{bmatrix}.$$

where each r' is a vector of integers. □

Lemma 2.25
Let $V \in \mathcal{H}$ with $\mathrm{lde}(V) \geq 1$. Then there exists $R \in \mathcal{S}$ such that $\mathrm{lde}(\overline{R}^\dagger V) = \mathrm{lde}(V) - 1$.
Proof
By inspection of Fig. 2 we see that for every $2\times2\times2$ pattern q there exists $R \in \mathcal{S}$ such that $\mathfrak{p}(\overline{R}) = q$. As a result, if $\mathfrak{p}(V)$ is a $2\times2\times2$ or a 2×4 pattern, then there exists $R \in \mathcal{S}$ such that $\overline{R}$ has a pattern finer than $\mathfrak{p}(V)$. By Corollary 2.23, $\mathfrak{p}(V)$ is in fact a $2\times2\times2$ row-pattern or a 2×4 row-pattern and thus there exists $R \in \mathcal{S}$ such that $\overline{R}$ is finer than V. We can then conclude by Lemma 2.24. □

Theorem 2.26
We have $\overline{\mathcal{G}} = \mathcal{H}$.
Proof
$\overline{\mathcal{G}} \subseteq \mathcal{H}$ by Proposition 2.13. We now show $\mathcal{H} \subseteq \overline{\mathcal{G}}$. Let $V \in \mathcal{H}$. We proceed by induction on the least denominator exponent of V. If $\mathrm{lde}(V) = 0$ then, by Lemma 2.20, $V \in \overline{\mathcal{C}}$ and therefore $V \in \overline{\mathcal{G}}$. Now if $\mathrm{lde}(V) > 0$, let R be the element of $\mathcal{S}$ with the lowest index such that $\mathrm{lde}(\overline{R}^\dagger V) = k - 1$. Such an element exists by Lemma 2.25. By the induction hypothesis we have $\overline{R}^\dagger V \in \overline{\mathcal{G}}$ which implies that $\overline{R}(\overline{R}^\dagger V) = V \in \overline{\mathcal{G}}$. □
The proof of Theorem 2.26 provides an algorithm to decompose an arbitrary element of $\overline{\mathcal{G}}$ into a product of elements of $\overline{\mathcal{S}}$, followed by an element of $\overline{\mathcal{C}}$. In the proof, there is freedom in choosing the element of $\overline{\mathcal{S}}$ used to reduce $\mathrm{lde}(\overline{V})$. If there is more than one generator with a finer pattern than $\overline{V}$, we must make a choice. The ordering imposed on $\mathcal{S}$ in Section "Generators" is used to make this choice in a uniform manner: we always choose the element of $\mathcal{S}$ of lowest index. As a result, the exact synthesis algorithm becomes deterministic. The ambiguity in the choice of generator is a consequence of the relations given in Lemma 2.2. In particular, we have

$$R(P,L)R(P,Q) = R(P,Q)R(P,iQL) = R(P,iQL)R(P,L)$$

and these three distinct sequences of generators denote the same operator. This is the source of the three-fold ambiguity in choosing a finer $2\times2\times2$ pattern for a given 2×4 pattern.
We will sometimes refer to the association between elements of $\mathcal{S}$ and patterns used in the exact synthesis algorithm of Theorem 2.26 as the *first finer partition* association, or FFP for short. The association is explicitly described Table 1.

Theorem 2.27
If U is a Clifford+CS operator such that $\mathrm{lde}(\overline{U}) = k$, then U can be represented by a Clifford+CS circuit of CS-count k. This circuit is optimal in CS-count and can be constructed in $\mathcal{O}(k)$ arithmetic operations.
Proof
Let U be as stated. If $k = 0$, then $\overline{U}$ belongs to $\overline{\mathcal{C}}$ and U is therefore a Clifford. If $k > 0$, then as in Theorem 2.26, there is a unique $R_k \in \mathcal{S}$ given by FFP such that $\mathrm{lde}(\overline{R}_k^\dagger \overline{U}) = k - 1$. By induction on the least denominator exponent, we have a deterministic synthesis algorithm to find a sequence such that

$$\overline{U} = \overline{R}_k \cdots \overline{R}_1 \cdot \overline{C}$$

which then implies that $U = R_k \cdots R_1 C$. Each of these k steps involves a constant number of basic arithmetic operations. This

Table 1. The elements of $\mathcal{S}$ and the explicit row patterns they are associated with under FFP.

Generator	Associated patterns under first finer partition (FFP)
$R(X\otimes I, I\otimes X)$	$\{\{1,4\},\{2,3\},\{5,6\}\},\{\{1,4\},\{2,3,5,6\}\},\{\{2,3\},\{1,4,5,6\}\},\{\{5,6\},\{1,2,3,4\}\}$
$R(Y\otimes I, I\otimes Y)$	$\{\{1,3\},\{2,5\},\{4,6\}\},\{\{1,3\},\{2,4,5,6\}\},\{\{2,5\},\{1,3,4,6\}\},\{\{4,6\},\{1,2,3,5\}\}$
$R(Z\otimes I, I\otimes Z)$	$\{\{1,2\},\{3,6\},\{4,5\}\},\{\{1,2\},\{3,4,5,6\}\},\{\{3,6\},\{1,2,4,5\}\},\{\{4,5\},\{1,2,3,6\}\}$
$R(Y\otimes I, I\otimes Z)$	$\{\{1,3\},\{2,6\},\{4,5\}\},\{\{2,6\},\{1,3,4,5\}\}$
$R(Z\otimes I, I\otimes Y)$	$\{\{1,2\},\{3,5\},\{4,6\}\},\{\{3,5\},\{1,2,4,6\}\}$
$R(Z\otimes I, I\otimes X)$	$\{\{1,2\},\{3,4\},\{5,6\}\},\{\{3,4\},\{1,2,5,6\}\}$
$R(X\otimes I, I\otimes Z)$	$\{\{1,6\},\{2,3\},\{4,5\}\},\{\{1,6\},\{2,3,4,5\}\}$
$R(X\otimes I, I\otimes Y)$	$\{\{1,5\},\{2,3\},\{4,6\}\},\{\{1,5\},\{2,3,4,6\}\}$
$R(Y\otimes I, I\otimes X)$	$\{\{1,3\},\{2,4\},\{5,6\}\},\{\{2,4\},\{1,3,5,6\}\}$
$R(X\otimes X, Y\otimes Y)$	$\{\{1,4\},\{2,5\},\{3,6\}\}$
$R(X\otimes X, Z\otimes Y)$	$\{\{1,4\},\{2,6\},\{3,5\}\}$
$R(Z\otimes X, Y\otimes Y)$	$\{\{1,6\},\{2,5\},\{3,4\}\}$
$R(Y\otimes X, X\otimes Y)$	$\{\{1,5\},\{2,4\},\{3,6\}\}$
$R(Z\otimes X, X\otimes Y)$	$\{\{1,5\},\{2,6\},\{3,4\}\}$
$R(Y\otimes X, Z\otimes Y)$	$\{\{1,6\},\{2,4\},\{3,5\}\}$

Table 2. Performance of the algorithm (in seconds) of Theorem 2.27 as implemented in our Mathematica code[38].

CS-count	Mean time (s)	Std. Dev. (s)
10	0.0138	0.0044
100	0.0281	0.0051
1000	0.1135	0.0091
10,000	1.1883	0.0897

Each run has constant overhead from computing the SO(6) representation for each unitary. Deviations from linearity are due to arithmetic operations on increasingly large integers. Each mean and standard deviation is computed using a sample of 1000 runs with pseudorandomly generated operators known to have the given minimal CS-count. Times are measured using Mathematica's in-built `AbsoluteTiming` function. Computations performed on a laptop with an Intel(R) Core(TM) i7 CPU running at 2.6 GHz with 6 cores and 16 GB of RAM runnning macOS Catalina version 10.15.7.

circuit has CS-count k, which is optimal by Lemma 2.15. □

Our Mathematica package[38] implements the algorithm referred to in Theorem 2.27 as well as a significant amount of other tools for two-qubit Clifford + CS circuits. Testing of the performance of this algorithm on a modest device is presented in Table 2.

Automata as tools for describing normal forms

In the previous section, we introduced a synthesis algorithm for Clifford+CS operators. The algorithm takes as input a Clifford+CS matrix and outputs a circuit for the corresponding operator. The circuit produced by the synthesis algorithm is a word over the alphabet $\mathcal{S}\cup\mathcal{C}$. Because the algorithm is deterministic, the word it associates to each operator can be viewed as a normal form for that operator. In the present section, we use the language of automata to give a detailed description of the structure of these normal forms. We include the definitions of some basic concepts from the theory of automata for completeness. The reader looking for further details is encouraged to consult[40].

In what follows we sometimes refer to a finite set Σ as an *alphabet*. In such a context, the elements of Σ are referred to as *letters*, Σ^* denotes the set of *words* over Σ (which includes the empty word ε), and the subsets of Σ^* are called *languages over* Σ. If $w\in\Sigma^*$ is a@@ word over the alphabet Σ, we write $|w|$ for the *length* of w. Finally, if L and L' are two languages over an alphabet Σ then their *concatenation* $L\circ L'$ is defined as $L\circ L'=\{ww'\,;\, w\in L \text{ and } w'\in L'\}$.

Definition 2.28

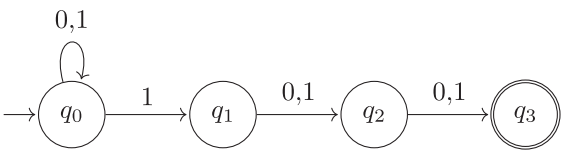
A *nondeterministic finite automaton* is a 5-tuple $(\Sigma, Q, \mathrm{In}, \mathrm{Fin}, \delta)$ where Σ and Q are finite sets, In and Fin are subsets of Q, and $\delta: Q\times(\Sigma\cup\{\varepsilon\}) :\to \mathscr{P}(Q)$ is a function whose codomain is the power set of Q. We call Σ the *alphabet*, Q the set of *states*, In and Fin the sets of *initial* and *final* states, and δ the *transition function*.

Remark 2.19

definition 2.28 is slightly non-standard. indeed, automata are typically defined as having a single initial state, rather than a collection of them. one can then think of definition 2.28 as introducing a collection of automata: one for each element of In. Alternatively, definition 2.28 can also be recovered from the usual definition by assuming that every automaton in the sense of definition 2.28 in fact has a single initial state s_0 related to the elements of In by $\delta(s_0,\varepsilon)=\mathrm{In}$. we chose to introduce automata as in definition 2.28 because this results in a slightly cleaner presentation. It is common to define an automaton $A=(\Sigma, Q, \mathrm{In}, \mathrm{Fin}, \delta)$ by specifying a directed labeled graph called the *state graph* of A. The vertices of the graph are labeled by states and there is an edge labeled by a letter $w\in\Sigma$ between vertices labeled q and q' if $q'\in\delta(q,w)$. The initial and final states are distinguished using arrows and double lines, respectively. For brevity, parallel edges are drawn only once, with their labels separated by a comma.

Example 2.30

The state graph for a nondeterministic finite automaton $A=(\Sigma, Q, \delta, \mathrm{In}, \mathrm{Fin})$ is depicted below.



Here, $Q=\{q_0,q_1,q_2,q_3\}$, $\Sigma=\{0,1\}$, the collection of initial states is $\mathrm{In}=\{q_0\}$, the collection of final states is $\mathrm{Fin}=\{q_3\}$, and we have, e.g., $\delta(q_0,1)=\{q_0,q_1\}$.

An automaton $A=(\Sigma, Q, \mathrm{In}, \mathrm{Fin}, \delta)$ can be used to specify a language $\mathcal{L}(A)\subseteq\Sigma^*$. Intuitively, $\mathcal{L}(A)$ is the collection of all the words over Σ that specify a well-formed walk along the state graph of A. The following definition makes this intuition more precise.

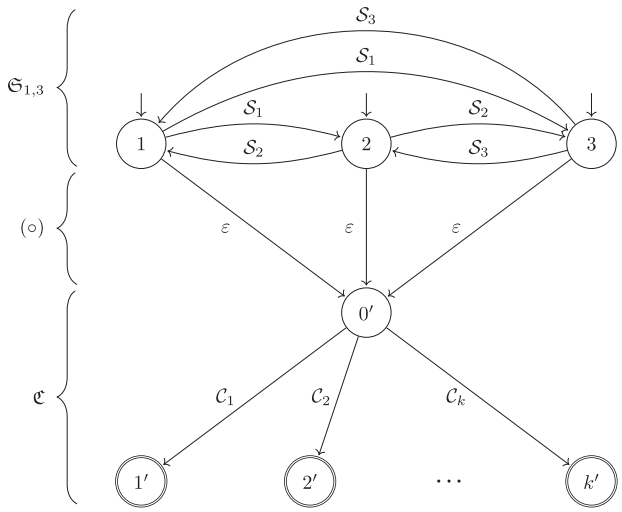


Fig. 3 The automaton $\mathfrak{S}_{1,3} \circ \mathfrak{C}$. The set of states of this automaton is $\{1, 2, 3, 0', 1', \ldots, k'\}$, which is the disjoint union of the states $\{1, 2, 3\}$ of $\mathfrak{S}_{1,3}$ and the states $\{0, 1, \ldots, k\}$ of $\mathfrak{C}$. The initial states are $\{1, 2, 3\}$, those of $\mathfrak{S}_{1,3}$, and the final states are $\{1', \ldots, k'\}$, those of $\mathfrak{C}$. Because $\mathfrak{S}_{1,3}$ has $\mathrm{Fin} = \{1, 2, 3\}$ and $\mathfrak{C}$ has $\mathrm{In} = \{0'\}$, the transition function δ of $\mathfrak{S}_{1,3} \circ \mathfrak{C}$ is such that $\delta(1, \varepsilon) = \delta(2, \varepsilon) = \delta(3, \varepsilon) = \{0'\}$. Otherwise, δ behaves like the transition function for $\mathfrak{S}_{1,3}$ on the subset of states $\{1, 2, 3\}$ and like the transition function for $\mathfrak{C}$ on the subset of states $\{0', 1', \ldots, k'\}$.

Definition 2.31
Let $A = (\Sigma, Q, \mathrm{In}, \mathrm{Fin}, \delta)$ be an automaton. Then *Aaccepts* a word $w = w_1 \cdots w_m \in \Sigma^*$ if there exists a sequence of states $s_0, s_1, \ldots, s_m \in Q$ such that

1. $s_0 \in \mathrm{In}$,
2. $s_{j+1} \in \delta(s_i, w_{j+1})$ for $j \in \{0, \ldots, m-1\}$, and
3. $s_m \in \mathrm{Fin}$.

The set of words accepted by A is called the language *recognized* by A and is denoted $\mathcal{L}(A)$.
Example 2.32 The alphabet for the automaton A given in Example 2.30 is $\Sigma = \{0, 1\}$. The language recognized by A is $\mathcal{L}(A) = \{w \in \Sigma^* ;\ \mathrm{thethirdrightmostletterof\ w\ is1}\}$.
If a language is recognized by some nondeterministic finite automata then that language is called *regular*. The collection of regular languages is closed under a variety of operations. In particular, regular languages are closed under concatenation.

Definition 2.33
Let $A = (\Sigma, Q, \mathrm{In}, \mathrm{Fin}, \delta)$ and $A' = (\Sigma, Q', \mathrm{In}', \mathrm{Fin}', \delta')$ be two automata. Then the *concatenation* of A and A' is the automaton $A \circ A' = (\Sigma, Q'', \mathrm{In}, \mathrm{Fin}', \delta'')$ where $Q'' = Q \sqcup Q'$ is the disjoint union of Q and Q' and

$$\delta''(q, s) = \begin{cases} \delta(q, s) & q \in Q \setminus \mathrm{Fin}, \\ \delta(q, s) & q \in \mathrm{Fin} \text{ and } s \neq \varepsilon, \\ \delta(q, s) \cup \mathrm{In}' & q \in \mathrm{Fin} \text{ and } s = \varepsilon, \text{ and} \\ \delta'(q, s) & q \in Q'. \end{cases}$$

Proposition 2.34
Let A and A' be automata recognizing languages L and L', respectively. Then $A \circ A'$ recognizes $L \circ L'$.
An example of the concatenation of two automata is provided in Fig. 3 and Example 2.38 based off of the automata defined in Definitions 2.36 and 2.37 below.

The structure of normal forms
We now consider the alphabet $\mathcal{S} \cup \mathcal{C}$ and describe the words over $\mathcal{S} \cup \mathcal{C}$ that are output by the synthesis algorithm of Theorem 2.27.

Definition 2.35
Let $U \in \mathcal{G}$. The *normal form of* U is the unique word over $\mathcal{S} \cup \mathcal{C}$ output by the synthesis algorithm of Theorem 2.27 on input U. We write $\mathcal{N}$ for the collection of all normal forms.
To describe the elements of $\mathcal{N}$, we introduce several automata. It will be convenient for our purposes to enumerate the elements of $\mathcal{C}$. We therefore assume that a total ordering of the 92160 elements of $\mathcal{C}$ is chosen and we write $\mathcal{C}_j$ for the j-th element of $\mathcal{C}$.

Definition 2.36
Let $k = |\mathcal{C}|$ and $\Sigma = \mathcal{S} \cup \mathcal{C}$. The automaton $\mathfrak{C}$ is defined as $\mathfrak{C} = (\Sigma, [0, k], \{0\}, [k], \delta_{\mathfrak{C}})$ where, for $s \in [0, k]$ and $\ell \in \Sigma$, we have

$$\delta_{\mathfrak{C}}(s, \ell) = \begin{cases} \{j\} & \text{if } s = 0 \text{ and } \ell = \mathcal{C}_j, \text{ and} \\ \varnothing & \text{otherwise.} \end{cases}$$

Definition 2.37
Let $\Sigma = \mathcal{S} \cup \mathcal{C}$. The automaton $\mathfrak{S}_{n,m}$ is defined as $\mathfrak{S}_{n,m} = (\Sigma, [m], [n, m], [m], \delta_{\mathfrak{S},m})$ where, for $s \in [m]$ and $\ell \in \Sigma$, we have

$$\delta_{\mathfrak{S},m}(s, \ell) = \begin{cases} \{t\,;\ \mathfrak{p}(\overline{\mathcal{S}_s}) \cap \mathfrak{p}(\overline{\mathcal{S}_t}) = \varnothing\} & \text{if } \ell = \mathcal{S}_s \text{ and} \\ \varnothing & \text{otherwise.} \end{cases}$$

Example 2.38. To illustrate Defiitions 2.33, 2.36, and 2.37, the automaton $\mathfrak{S}_{1,3} \circ \mathfrak{C}$ is represented in Fig. 3. It can be verified that the words $\mathcal{C}_2$, $\mathcal{S}_2\mathcal{S}_1\mathcal{C}_1$, and $\mathcal{S}_3\mathcal{S}_1\mathcal{S}_2\mathcal{C}_k$ are accepted by $\mathfrak{S}_{1,3} \circ \mathfrak{C}$ while the words $\mathcal{S}_1\mathcal{S}_1\mathcal{C}_4$ and $\mathcal{S}_3\mathcal{C}_7\mathcal{S}_1$ are not. Note in particular that if $\mathcal{C}_1$ is the symbol for the identity, then $\mathcal{S}_3\mathcal{C}_1$ is distinct (as a word) from $\mathcal{S}_3$. The former is accepted by $\mathfrak{S}_{1,3} \circ \mathfrak{C}$ while the latter is not. Despite the state graph of $\mathfrak{S}_{1,3}$ being fully-connected, full-connectivity does not necessarily hold for state graphs of other $\mathfrak{S}_{n,m}$ automata.
We will use the automata introduced in Definitions 2.36 and 2.37 to describe the elements of $\mathcal{N}$. Our goal is to show that

$$\mathcal{N} = \mathcal{L}(\mathfrak{S}_{1,3} \circ \mathfrak{S}_{4,9} \circ \mathfrak{S}_{10,15} \circ \mathfrak{C}) \tag{17}$$

We start by establishing a few propositions.

Proposition 2.39
We have $\mathcal{L}(\mathfrak{C}) \subsetneq \mathcal{L}(\mathfrak{S}_{1,15} \circ \mathfrak{C}) \subsetneq \mathcal{L}(\mathfrak{S}_{1,9} \circ \mathfrak{S}_{10,15} \circ \mathfrak{C}) \subsetneq \mathcal{L}(\mathfrak{S}_{1,3} \circ \mathfrak{S}_{4,9} \circ \mathfrak{S}_{10,15} \circ \mathfrak{C})$, where $\subsetneq$ denotes strict inclusion.
Proof
By Definitions 2.36 and 2.37. □
We emphasize that the inclusions in Proposition 2.39 are strict. This implies that $\mathcal{L}(\mathfrak{S}_{1,3} \circ \mathfrak{S}_{4,9} \circ \mathfrak{S}_{10,15} \circ \mathfrak{C})$ can be written as the disjoint union of $\mathcal{L}(\mathfrak{C})$, $\mathcal{L}(\mathfrak{S}_{1,15} \circ \mathfrak{C})$, and $\mathcal{L}(\mathfrak{S}_{1,9} \circ \mathfrak{S}_{10,15} \circ \mathfrak{C})$. The lemmas below show that these languages correspond to disjoint subsets of $\mathcal{N}$ and, in combination, suffice to prove Eq. (17).

Lemma 2.40
Let U be a word over $\mathcal{S} \cup \mathcal{C}$. Then $U \in \mathcal{L}(\mathfrak{C})$ if and only if $U \in \mathcal{N}$ and U has length 1, i.e, $U \in \mathcal{C}$.
Proof
By Definition 2.36 and Theorem 2.27. □

Lemma 2.41
Let U be a word over $\mathcal{S} \cup \mathcal{C}$. Then $U \in \mathcal{L}(\mathfrak{S}_{1,15} \circ \mathfrak{C}) \setminus \mathcal{L}(\mathfrak{C})$ if and only if $U \in \mathcal{N}$ and U has a $2 \times 2 \times 2$ pattern.
Proof
First, note that $\mathcal{L}(\mathfrak{C})$ is the set of words of length 1 accepted by $\mathfrak{S}_{1,15} \circ \mathfrak{C}$. This means that $\mathcal{L}(\mathfrak{S}_{1,15} \circ \mathfrak{C}) \setminus \mathcal{L}(\mathfrak{C})$ consists of all the words of length $k \geq 2$ accepted by $\mathfrak{S}_{1,15} \circ \mathfrak{C}$. Furthermore, by Lemma 2.20, there are no normal forms of length 1 which have a

$2\times2\times2$ pattern. Thus, to prove our lemma it suffices to establish the following equality of sets

$$\{U\in\mathcal{L}(\mathfrak{S}_{1,15}\circ\mathfrak{C})\,;\,|U|=k\}=\{U\in\mathcal{N}\,;\\ |U|=k \text{ and } \mathfrak{p}(U) \text{ is a} 2\times2\times2 \text{ pattern}\} \tag{18}$$

for all $k\geq2$. We proceed by induction on k.

- Note that, by definition of $\mathfrak{S}_{1,15}\circ\mathfrak{C}$, we have $\{U\in\mathcal{L}(\mathfrak{S}_{1,15}\circ\mathfrak{C})\,;\,|U|=2\}=\mathcal{SC}$. Every element of $\mathcal{SC}$ has a $2\times2\times2$ pattern by Lemma 2.21. Moreover, for $U=SC$ with $S\in\mathcal{S}$ and $C\in\mathcal{C}$, $\mathfrak{p}(SC)=\mathfrak{p}(S)$. Thus, SC must also be the unique word produced by the synthesis algorithm on input U and hence $U\in\mathcal{N}$. This accounts for all words of length 2 in $\mathcal{N}$. Therefore Eq. (18) holds when $k=2$.
- Now suppose that Eq. (18) holds for some $k\geq2$. Let $U\in\mathcal{L}(\mathfrak{S}_{1,15}\circ\mathfrak{C})$ be a word of length k whose first letter is $S\in\mathcal{S}$. Then $U\in\mathcal{N}$ and $\mathfrak{p}(U)=\mathfrak{p}(S)$ is a $2\times2\times2$ pattern. Furthermore, the least denominator exponent of $\overline{U}$ is $k-1$. We will show that Eq. (18) holds for $k+1$ by establishing two inclusions. Because it will sometimes be convenient to refer to submatrices, if M is an $n\times n$ matrix and $x,y\subseteq[n]$, we write

  $$M[x;y]$$

  for the submatrix of M formed from the rows with index in x and the columns with index in y.

$\subseteq$: Suppose that $U'=S'U$ is a word of length $k+1$ accepted by $\mathcal{L}(\mathfrak{S}_{1,15}\circ\mathfrak{C})$. Then by Definition 2.37 we have $\mathfrak{p}(S')\cap\mathfrak{p}(S)=\varnothing$. Let $\{a,b\}\in\mathfrak{p}(S')$, and let r_a and r_b be the corresponding rows of the residue matrix of $\overline{U}$. Explicitly, we have

$$\rho_{k-1}(\overline{U})[\{a,b\};[6]]=\begin{bmatrix}r_a\\r_b\end{bmatrix}$$

with $r_a\neq r_b$ as $\{a,b\}$ is not a subset of any element of $\mathfrak{p}(U)$. Direct calculation of the rows of the residue matrix for $\overline{U'}$ yields

$$\rho_k(\overline{U'})[\{a,b\};[6]]=\begin{bmatrix}r_a+r_b\\r_a+r_b\end{bmatrix}.$$

We conclude that $\{a,b\}$ is a subset of an element of $\mathfrak{p}(U')$. Furthermore, by Lemma 2.22 and Eq. (16) we see that, since $r_a+r_b\neq0$, $\mathfrak{p}(U')$ cannot be a 2×4 pattern, and therefore $\{a,b\}\in\mathfrak{p}(U')$. As this holds for all $\{a,b\}\in\mathfrak{p}(S')$, we conclude that $\mathfrak{p}(S')=\mathfrak{p}(U')$. Thus, by the induction hypothesis, $S'U$ will be the word produced by the synthesis algorithm when applied to U'. Hence, $U'\in\mathcal{N}$ and $\mathfrak{p}(U')$ is a $2\times2\times2$ pattern.

$\supseteq$: Suppose that U' is a normal form of length $k+1$ with a $2\times2\times2$ pattern. Write U' as $U'=S'V$ for some unknown normal form V. We then have $\mathfrak{p}(S')=\mathfrak{p}(U')$. Let $\{a,b\}\in\mathfrak{p}(S')$ and let the corresponding rows of the residue matrix of $\overline{V}$ be r_a and r_b. Explicitly, we have

$$\rho_{k-1}(\overline{V})[\{a,b\};[6]]=\begin{bmatrix}r_a\\r_b\end{bmatrix}.$$

Direct calculation of the rows of the residue matrix for $\overline{U'}$ yields

$$\rho_k(\overline{U'})[\{a,b\};[6]]=\begin{bmatrix}r_a+r_b\\r_a+r_b\end{bmatrix}.$$

Since $\mathfrak{p}(U')$ is not a 2×4 pattern, we conclude that $r_a+r_b\neq0$ and thus that $r_a\neq r_b$. Therefore, there is no element of cardinality four in $\mathfrak{p}(V)$. Since $\mathrm{lde}(V)>0$, $\mathfrak{p}(V)$ must then be a $2\times2\times2$ pattern. Consequently, we have $V=U$ as defined above. Because $\{a,b\}\notin\mathfrak{p}(U)=\mathfrak{p}(S)$, we know $\mathfrak{p}(S')\cap\mathfrak{p}(S)=\varnothing$. Given that $S'=\mathcal{S}_{j'}$ and $S=\mathcal{S}_j$, we conclude that $j\in\delta_{\mathfrak{S},15}(j',S'=\mathcal{S}_{j'})$. Because $S=\mathcal{S}_j$ is the first letter of the word U, we know the initial state of U must be j. Therefore, by the induction hypothesis, $U'=S'U$ is accepted by $\mathfrak{S}_{1,15}\circ\mathfrak{C}$.

We have shown that Eq. (18) holds for words of length $k+1$ if it holds for words of length k. This completes the inductive step. □

Lemma 2.41 characterized the normal forms that have a $2\times2\times2$ pattern. The two lemmas below jointly characterize the normal forms that have a 2×4 pattern. Because their proofs are similar in spirit to that of Lemma 2.41, they have been relegated to Supplementary Note 3.

Lemma 2.42
Let U be a word over $\mathcal{S}\cup\mathcal{C}$. Then $U\in\mathcal{L}(\mathfrak{S}_{1,9}\circ\mathfrak{S}_{10,15}\circ\mathfrak{C})\setminus\mathcal{L}(\mathfrak{S}_{1,15}\circ\mathfrak{C})$ if and only if $U\in\mathcal{N}$ and U has a 2×4 pattern with $\mathfrak{p}(U)\cap\{\{x,y\}\,;\,(x,y)\in[3]\times[4,6]\}\neq\varnothing$.

Lemma 2.43
Let U be a word over $\mathcal{S}\cup\mathcal{C}$. Then $U\in\mathcal{L}(\mathfrak{S}_{1,3}\circ\mathfrak{S}_{4,9}\circ\mathfrak{S}_{10,15}\circ\mathfrak{C})\setminus\mathcal{L}(\mathfrak{S}_{1,9}\circ\mathfrak{S}_{10,15}\circ\mathfrak{C})$ if and only if $U\in\mathcal{N}$ and U has a 2×4 pattern with $\mathfrak{p}(U)\cap\{\{x,y\}\,;\,(x,y)\in([3],[4,6])\}=\varnothing$.

Theorem 2.44
Let U be a word over $\mathcal{S}\cup\mathcal{C}$. Then $U\in\mathcal{L}(\mathfrak{S}_{1,3}\circ\mathfrak{S}_{4,9}\circ\mathfrak{S}_{10,15}\circ\mathfrak{C})$ if and only if $U\in\mathcal{N}$.

Proof
If $|U|=1$ then the result follows from Lemma 2.40. If $|U|>1$, then U has a $2\times2\times2$ or a 2×4 pattern and the result follows from Proposition 2.39 and Lemmas 2.41, 2.42 and 2.43. □

Lower bounds

Recall that the *distance* between operators U and V is defined as $\|U-V\|=\sup\{\|Uv-Vv\|\,;\,\|v\|=1\}$. Because $\mathcal{G}$ is universal, for every $\epsilon>0$ and every element $U\in\mathrm{SU}(4)$, there exists $V\in\mathcal{G}$ such that $\|U-V\|\leq\epsilon$. In such a case we say that V is an *ϵ-approximation* of U. We now take advantage of Theorem 2.44 to count Clifford+CS operators and use these results to derive a worst-case lower bound on the CS-count of approximations.

Lemma 2.45
Let $n\geq1$. There are $86400(3\cdot8^n-2\cdot4^n)$ Clifford+CS operators of CS-count exactly n.

Proof
Each Clifford+CS operator is represented by a unique normal form and this representation is CS-optimal. Hence, to count the number of Clifford+CS operators of CS-count n, it suffices to count the normal forms of CS-count n. By Theorem 2.44, and since Clifford operators have CS-count 0, a normal form of CS-count n is a word

$$w=w_1w_2w_3w_4 \tag{19}$$

such that $w_1\in\mathcal{L}(\mathfrak{S}_{1,3})$, $w_2\in\mathcal{L}(\mathfrak{S}_{4,9})$, $w_3\in\mathcal{L}(\mathfrak{S}_{10,15})$, $w_4\in\mathcal{L}(\mathfrak{C})$ and the CS-counts of w_1, w_2, and w_3 sum to n. There are

$$(6\cdot8^{n-1}+6\cdot4^{n-1}+3\cdot2^{n-1})\cdot|\mathcal{C}| \tag{20}$$

words of the form of Eq. (19) such that exactly one of w_1, w_2, or w_3 is not ε. Similarly, there are

$$\left(\sum_{0<l<n}18\cdot2^{2n-3-l}+\sum_{0<l<n}18\cdot2^{3n-4-2l}+\sum_{0<j<n}36\cdot2^{3n-5-j}\right)\cdot|\mathcal{C}| \tag{21}$$

words of the form of Eq. (19) such that exactly two of w_1, w_2, or w_3 are not ε. Finally, the number of words of the form of Eq. (19) such that w_1, w_2, and w_3 are not ε is

$$\left(\sum_{0<l<n-j}\sum_{0<j<n}108\cdot2^{3n-6-j-2l}\right)\cdot|\mathcal{C}| \tag{22}$$

Summing Eqs. (20), (21) and (22) and applying the geometric series formula then yields the desired result. □

Corollary 2.46
For $n \in \mathbb{N}$, there are $\frac{46080}{7}(45 \cdot 8^n - 35 \cdot 4^n + 4)$ distinct Clifford +CS operators of CS-count at most n.
Proof
Recall that the Clifford+CS operators of CS-count 0 are exactly the Clifford operators and that $|\mathcal{C}| = 92160$. The result then follows from Lemma 2.45 and the geometric series formula.

Proposition 2.47
For every $\epsilon \in \mathbb{R}^{>0}$, there exists $U \in \mathrm{SU}(4)$ such that any Clifford +CSϵ-approximation of U has CS-count at least $5\log_2(1/\epsilon) - 0.67$.
Proof
By a volume counting argument. Each operator must occupy an ϵ-ball worth of volume in 15-dimensional SU(4) space, and the sum of all these volumes must add to the total volume of SU(4) which is $(\sqrt{2}\pi^9)/3$. The number of circuits up to CS-count n is taken from Corollary 2.46 (we must divide the result by two to account for the absence of overall phase ω in the special unitary group) and a 15-dimensional ϵ-ball has a volume of

$$\frac{\pi^{\frac{15}{2}}}{\Gamma\left(\frac{15}{2}+1\right)}\epsilon^{15}.$$

Let U be an element of $\mathcal{G}$ of determinant 1. By Eq. (1) of Section "Generators", U can be written as

$$U = \frac{1}{\sqrt{2}^k}M$$

where $k \in \mathbb{N}$ and the entries of M belong to $\mathbb{Z}[i]$. We can therefore talk about the least denominator exponent of the SU(4) representation of U. We finish this section by relating the least denominator exponent of the SU(4) representation of U and the CS-count of the normal form of U.

Proposition 2.48
Let U be an element of $\mathcal{G}$ of determinant 1, let k be the least denominator exponent of the SU(4) representation of U, and let k' be the CS-count of the normal form of U. Then

$$\frac{k-3}{2} \leq k' \leq 2k+2.$$

Proof
The CS-count of the normal form of U is equal to the least denominator exponent of the SO(6) representation of U. Eq. (11) then implies the upper bound for k'. Likewise, examination of Theorem 2.44 reveals that the CS operators in the circuit for U must be separated from one another by a Clifford with a least denominator exponent of at most 2 in its unitary representation. Combining this with the fact that the largest least denominator exponent of an operator in $\mathcal{C}$ is 3, we arrive at the lower bound for k'. □
Remark 2.49
It was established in ref. [8] that, for single-qubit Clifford+T operators of determinant 1, there is a simple relation between the least denominator exponent of an operator and its T-count: if the least denominator exponent of the operator is k, then its T-count is $2k-2$ or $2k$. Interestingly, this is not the case for Clifford +CS operators in SU(4), as suggested by Proposition 2.48. Clearly, the CS-count of an operator always scales linearly with the least denominator exponent of its unitary representation. For large k, computational experiments with our code[38] suggest that most operators are such that $k' \approx k$, though there are examples of operators with $k' \approx 2k$. One example of such an operator is $[R(X\otimes I, I\otimes Z)R(X\otimes I, I\otimes X)R(Z\otimes I, I\otimes X)R(Z\otimes I, I\otimes Z)]^m$ for $m \in \mathbb{N}$.

DISCUSSION

We described an exact synthesis algorithm for a fault-tolerant multi-qubit gate set which is simultaneously optimal, practically efficient, and explicitly characterizes all possible outputs. The algorithm establishes the existence of a unique normal form for two-qubit Clifford+CS circuits. We showed that the normal form for an operator can be computed with a number of arithmetic operations linear in the gate-count of the output circuit. Finally, we used a volume counting argument to show that, in the typical case, ϵ-approximations of two-qubit unitaries will require a CS-count of at least $5\log_2(1/\epsilon)$.

We hope that the techniques developed in the present work can be used to obtain optimal multi-qubit normal forms for other two-qubit gate sets, such as the two-qubit Clifford+T-gate set. Indeed, it can be shown that the SO(6) representation of Clifford+T operators are exactly the set of SO(6) matrices with entries in the ring $\mathbb{Z}[1/\sqrt{2}]$. Further afield, the exceptional isomorphism for SU(8) could potentially be leveraged to design good synthesis algorithms for three-qubit operators. Such algorithms would provide a powerful basis for more general quantum compilers.

An interesting avenue for future research is to investigate whether the techniques and results presented in this paper can be used in the context of *synthillation*. Quantum circuit synthesis and magic state distillation are often kept separate. But it was shown in ref. [41] that performing synthesis and distillation simultaneously (synthillation) can lead to overall savings. The analysis presented in ref. [41] uses T gates and T states. Leveraging higher-dimensional synthesis methods such as the ones presented here, along with distillation of CS states, could yield further savings.

METHODS

All results were produced theoretically or computationally, with the requisite methods described at length in each section.

DATA AVAILABILITY

The sets of various CS-count operators used to generate the algorithmic performance information in Table 2 are available at ref. [38].

CODE AVAILABILITY

The Mathematica package referenced throughout the paper and its documentation are publically available from the repository at ref. [38].

Received: 29 June 2020; Accepted: 2 May 2021;
Published online: 22 June 2021

REFERENCES

1. Reichardt, B. W. Quantum universality from magic states distillation applied to CSS codes. *Quantum Inf. Process.* **4**, 251–264 (2005).
2. Blass, A., Bocharov, A. & Gurevich, Y. Optimal ancilla-free Pauli+V circuits for axial rotations. *J. Math. Phys.* **56**, 122201 (2014).
3. Bocharov, A., Gurevich, Y. & Svore, K. M. Efficient decomposition of single-qubit gates into V basis circuits. *Phys. Rev. A* **88**, 012313 (2013).
4. Forest, S., Gosset, D., Kliuchnikov, V. & McKinnon, D. Exact synthesis of single-qubit unitaries over Clifford-cyclotomic gate sets. *J. Math. Phys.* **56**, 082201 (2015).
5. Kliuchnikov, V., Maslov, D. & Mosca, M. Fast and efficient exact synthesis of single-qubit unitaries generated by Clifford and T gates. *Quantum Inf. Comput.* **13**, 607–630 (2013).
6. Kliuchnikov, V. & Yard, J. A framework for exact synthesis (2015). Preprint.
7. Ross, N. J. Optimal ancilla-free Clifford+V approximation of z-rotations. *Quantum Inf. Comput.* **15**, 932–950 (2015).
8. Ross, N. J. & Selinger, P. Optimal ancilla-free Clifford+T approximation of z-rotations. *Quantum Inf. Comp.* **16**, 901–953 (2016).
9. Bocharov, A., Cui, X., Kliuchnikov, V. & Wang, Z. Efficient topological compilation for a weakly integral anyonic model. *Phys. Rev. A* **93**, 012313 (2016).

10. Glaudell, A. N., Ross, N. J. & Taylor, J. M. Canonical forms for single-qutrit Clifford +*T* operators. *Ann. Phys.* **406**, 54–70 (2019).
11. Kliuchnikov, V., Bocharov, A. & Svore, K. M. Asymptotically optimal topological quantum compiling. *Phys. Rev. Lett.* **112**, 140504 (2014).
12. Prakash, S., Jain, A., Kapur, B. & Seth, S. Normal form for single-qutrit Clifford+*T* operators and synthesis of single-qutrit gates. *Phys. Rev. A* **98**, 032304 (2018).
13. Kliuchnikov, V., Bocharov, A., Roetteler, M. & Yard, J. A framework for approximating qubit unitaries (2015). Preprint.
14. Kliuchnikov, V., Maslov, D. & Mosca, M. Practical approximation of single-qubit unitaries by single-qubit quantum Clifford and *T* circuits. *IEEE T. Comput.* **65**, 161–172 (2016).
15. Shende, V. V., Markov, I. L. & Bullock, S. S. Minimal universal two-qubit controlled-NOT-based circuits. *Phys. Rev. A* **69**, 062321 (2004).
16. Zhang, J., Vala, J., Sastry, S. & Whaley, K. B. Geometric theory of nonlocal two-qubit operations. *Phys. Rev. A* **67**, 042313 (2003).
17. Amy, M., Glaudell, A. N. & Ross, N. J. Number-theoretic characterizations of some restricted Clifford+*T* circuits. *Quantum* **4**, 252 (2020).
18. Amy, M., Maslov, D. & Mosca, M. Polynomial-time *T*-depth optimization of Clifford +*T* circuits via matroid partitioning. *IEEE T. Comput. Aid. D.* **33**, 1476–1489 (2014).
19. Amy, M., Maslov, D., Mosca, M. & Roetteler, M. A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits. *IEEE T. Comput. Aid. D.* **32**, 818–830 (2013).
20. Giles, B. & Selinger, P. Exact synthesis of multiqubit Clifford+*T* circuits. *Phys. Rev. A* **87**, 032332 (2013).
21. Gosset, D., Kliuchnikov, V., Mosca, M. & Russo, V. An algorithm for the *T*-count. *Quantum Inf. Comput.* **14**, 1261–1276 (2014).
22. Heyfron, L. E. & Campbell, E. T. An efficient quantum compiler that reduces *T*-count. *Quantum Sci. Technol.* **4**, 015004 (2018).
23. Kissinger, A. & van de Wetering, J. Reducing the number of non-Clifford gates in quantum circuits. *Phys. Rev. A* **102**, 022406 (2020).
24. Matteo, O. D. & Mosca, M. Parallelizing quantum circuit synthesis. *Quantum Sci. Technol.* **1**, 015003 (2016).
25. Meuli, G., Soeken, M. & Micheli, G. D. SAT-based {CNOT, *T*} quantum circuit synthesis. In *Lecture Notes in Computer Science, RC '17*, 175–188 (Springer International Publishing, 2018).
26. Mosca, M. & Mukhopadhyay, P. A polynomial time and space heuristic algorithm for *T*-count (2020). Preprint.
27. Zhang, F. & Chen, J. Optimizing *T* gates in Clifford+*T* circuit as $\pi/4$ rotations around Paulis (2019). Preprint.
28. Giles, B. & Selinger, P. Remarks on Matsumoto and Amano's normal form for single-qubit Clifford+*T* operators (2013). Preprint.
29. Selinger, P. Efficient clifford+*T* approximation of single-qubit operators. *Quantum Inf. Comput.* **15**, 159–180 (2015).
30. Beverland, M., Campbell, E., Howard, M. & Kliuchnikov, V. Lower bounds on the non-Clifford resources for quantum computations. *Quantum Sci. Technol.* **5**, 035009 (2020).
31. Haah, J. & Hastings, M. B. Codes and protocols for distilling *T*, controlled-*S*, and Toffoli gates. *Quantum* **2**, 71 (2018).
32. Cross, A. W. et al. Scalable randomised benchmarking of non-Clifford gates. *npj Quantum Inf.* **2**, 1–5 (2016).
33. Garion, S. & Cross, A. W. Synthesis of CNOT-dihedral circuits with optimal number of two qubit gates. *Quantum* **4**, 369 (2020).
34. Garion, S. et al. Experimental implementation of non-Clifford interleaved randomized benchmarking with a controlled-$\mathcal{S}$ gate. *Phys. Rev. Res.* **3**, 013204 (2021).
35. Sheldon, S., Magesan, E., Chow, J. M. & Gambetta, J. M. Procedure for systematically tuning up cross-talk in the cross-resonance gate. *Phys. Rev. A* **93**, 060302 (2016).
36. Foxen, B. et al. Demonstrating a continuous set of two-qubit gates for near-term quantum algorithms. *Phys. Rev. Lett.* **125**, 120504 (2020).
37. Xu, Y. et al. High-fidelity, high-scalability two-qubit gate scheme for superconducting qubits. *Phys. Rev. Lett.* **125**, 240503 (2020).
38. Glaudell, A. N., Ross, N. J. & Taylor, J. M. GaussSynth. https://doi.org/10.5281/zenodo.4549819. (2020).
39. Tilma, T. & Sudarshan, E. Generalized Euler angle parametrization for SU(*N*). *J. Phys. A Math. Gen.* **35**, 10467 (2002).
40. Sipser, M. *Introduction to the Theory of Computation,* 1st edn (International Thomson Publishing, 1996).
41. Campbell, E. T. & Howard, M. Unified framework for magic state distillation and multiqubit gate synthesis with reduced resource cost. *Phys. Rev. A* **95**, 022316 (2017).

ACKNOWLEDGEMENTS

A.N.G. was partially supported by the Princeton Center for Complex Materials, a MRSEC supported by NSF grant DMR 1420541. N.J.R. was partially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), funding reference number RGPIN-2018-04064. We would like to thank Matthew Amy, Xiaoning Bian, and Peter Selinger for helpful discussions. In addition, we would like to thank the anonymous reviewers whose comments greatly improved the paper.

AUTHOR CONTRIBUTIONS

All authors researched, collated, and wrote this paper.

COMPETING INTERESTS

The authors declare no competing interests.

ADDITIONAL INFORMATION

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41534-021-00424-z.

Correspondence and requests for materials should be addressed to A.N.G.

Reprints and permission information is available at http://www.nature.com/reprints

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.