Logical Clifford Synthesis for Stabilizer Codes

NARAYANAN RENGASWAMY[1] (Student Member, IEEE),
ROBERT CALDERBANK[1] (Fellow, IEEE),
SWANAND KADHE[2] (Member, IEEE),
AND HENRY D. PFISTER[1] (Senior Member, IEEE)

[1] Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708 USA
[2] Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720 USA

Corresponding author: Narayanan Rengaswamy (e-mail: narayanan.rengaswamy@duke.edu).

ABSTRACT Quantum error-correcting codes are used to protect qubits involved in quantum computation. This process requires logical operators to be translated into physical operators acting on physical quantum states. In this article, we propose a mathematical framework for synthesizing physical circuits that implement logical Clifford operators for stabilizer codes. Circuit synthesis is enabled by representing the desired physical Clifford operator in $\mathbb{C}^{N \times N}$ as a $2m \times 2m$ binary symplectic matrix, where $N = 2^m$. We prove two theorems that use symplectic transvections to efficiently enumerate all binary symplectic matrices that satisfy a system of linear equations. As a corollary, we prove that for an $[\![m, k]\!]$ stabilizer code every logical Clifford operator has $2^{r(r+1)/2}$ symplectic solutions, where $r = m - k$, up to stabilizer degeneracy. The desired physical circuits are then obtained by decomposing each solution into a product of elementary symplectic matrices, that correspond to elementary circuits. This enumeration of all physical realizations enables optimization over the ensemble with respect to a suitable metric. Furthermore, we show that any circuit that normalizes the stabilizer can be transformed into a circuit that centralizes the stabilizer, while realizing the same logical operation. Our method of circuit synthesis can be applied to any stabilizer code, and this paper discusses a proof of concept synthesis for the $[\![6, 4, 2]\!]$ CSS code. Programs implementing the algorithms in this article, which includes routines to solve for binary symplectic solutions of general linear systems and our overall LCS (logical circuit synthesis) algorithm, can be found at https://github.com/nrenga/symplectic-arxiv18a

INDEX TERMS Clifford group, Heisenberg-Weyl group, logical operators, stabilizer codes, binary symplectic group, transvections.

I. INTRODUCTION

It is expected that universal fault-tolerant quantum computation will be achieved by employing Quantum Error-Correcting Codes (QECCs) to protect the information stored in the quantum computer and to enable error-resilient computation on that data. The first QECC was discovered by Shor [2], and subsequently, a systematic framework was developed by Calderbank, Shor and Steane [3], [4] to translate (pairs of) classical error-correcting codes into QECCs. Codes produced using this framework are referred to as *CSS* codes. The general class of *stabilizer* codes includes CSS codes as a special case and was introduced by Calderbank, Rains, Shor and Sloane [5], and by Gottesman [6]. These codes, and their variations [7], [8], still remain the preferred class of codes for realizing error-resilient quantum computation in practice.

The *Clifford hierarchy* of unitary operators was defined to help demonstrate that universal quantum computation can be realized via the teleportation protocol [9]. The first level $\mathcal{C}^{(1)}$ in the hierarchy is the Pauli group of unitary operators, and subsequent levels $\mathcal{C}^{(\ell)}$, $\ell \geq 2$, are defined recursively as those unitary operators that map the Pauli group into $\mathcal{C}^{(\ell-1)}$, under conjugation. By this definition, the second level is the normalizer of the Pauli group in the unitary group, and hence $\mathcal{C}^{(2)}$ is the Clifford group [5]. It is well-known that the levels $\mathcal{C}^{(\ell)}$ do not form a group for $\ell \geq 3$, but that the Clifford group along with *any* unitary in $\mathcal{C}^{(3)}$ can be used to approximate an arbitrary unitary operator up to any desired precision. (Note that using a simple inductive argument it can be proven that each level in the hierarchy is closed under multiplication by Clifford group elements.) Therefore, the standard strategy for realizing universal computation with

QECCs is to first synthesize[1] logical Paulis, then logical Cliffords, and finally some logical non-Clifford in the third level of the Clifford hierarchy. In this article, we will be primarily concerned with logical Cliffords because specific QECCs, such as *tri-orthogonal codes* [10], can be used to distill *magic states* [11] for a non-Clifford gate in $\mathcal{C}^{(3)}$, and these states can then be "injected" into the computation via teleportation in order to realize the action of that gate at the logical level [9]. Hence, any circuit implemented on the computer equipped with error-correction might be expected to consist only of Clifford gates, augmented with ancilla magic states, and Pauli measurements.

For the task of synthesizing the logical Pauli operators for stabilizer codes, the first algorithm was introduced by Gottesman [6, Sec. 4] and subsequently, another algorithm based on a symplectic Gram-Schmidt procedure was proposed by Wilde [12]. The latter is closely related to earlier work by Brun *et al.* [13], [14]. Since the logical Paulis are inputs to our algorithm that synthesizes logical Clifford operators for stabilizer codes, we will consider the above two procedures to be "preprocessors" for our algorithm.

Given the logical Pauli operators for an $[\![m, k]\!]$ stabilizer QECC, that encodes k logical qubits into m physical qubits, physical Clifford realizations of Clifford operators on the logical qubits can be represented by $2m \times 2m$ binary symplectic matrices; thereby, reducing the complexity *dramatically* from 2^{2m} complex variables to $4m^2$ binary variables (see [15], [16] and Section II). We exploit this fact to propose an algorithm that efficiently assembles *all* $2^{r(r+1)/2}$, where $r = m - k$, symplectic matrices representing physical Clifford operators (circuits) that realize a given logical Clifford operator on the protected qubits. We will refer to this procedure as the *Logical Clifford Synthesis (LCS) algorithm*. Here, each symplectic solution represents an equivalence class of Clifford circuits, all of which "propagate" input Pauli operators through them in an identical fashion (see Section III). Moreover, as we will discuss later in the context of the algorithm, the other degrees of freedom not captured by our algorithm are those provided by stabilizers (see Remark 12). However, at the cost of some increased computational complexity, the algorithm can easily be modified to account for these stabilizer degrees of freedom. Hence, this article makes it possible to optimize the choice of circuit with respect to a suitable metric, that might be a function of the quantum hardware. Note that our approach here is to determine *unitary* physical operations to realize a specific logical (Clifford) operation, and this is distinct from operations, such as lattice surgery [17] that are used to perform logical operations on topological codes.

The primary contributions of this paper are the four theorems that we state and prove in Section III-B, and the main

LCS algorithm (Algorithm 3), which builds on the results of these theorems. These results form part of a larger program for fault-tolerant quantum computation, where the goal is to achieve reliability by using classical computers to track and control physical quantum systems, and perform error correction only as needed.

We note that there are several works that focus on exactly decomposing, or approximating, an arbitrary unitary operator as a sequence of operators from a fixed *instruction set*, such as Clifford $+ T$ [18]–[23]. However, these works do not consider the problem of circuit synthesis or optimization over different realizations of unitary operators on the *encoded* space. We also note that there exists several works in the literature that study this problem for specific codes and operations, e.g., see [6]–[8], [24]–[27]. However, we believe this article is the first to propose a systematic framework to address this problem for general stabilizer codes, and hence enable automated circuit synthesis for encoded Clifford operators. This procedure is more systematic in considering all degrees of freedom than conjugating the desired logical operator by the encoding circuit for the QECC.

Recently, we have used the LCS algorithm to translate the unitary 2-design we constructed from classical Kerdock codes into a *logical* unitary 2-design [28], and in general, any design consisting of only Clifford elements can be transformed into a logical design using our algorithm. An implementation of the design is available on github.[2] This finds direct application in the *logical randomized benchmarking* protocol proposed by Combes *et al.* [29]. This protocol is a more robust procedure to estimate logical gate fidelities than extrapolating results from randomized benchmarking performed on physical gates [30]. Now, we discuss some more motivations and potential applications for the LCS algorithm.

A. NOISE VARIATION IN QUANTUM SYSTEMS

Although depth or the number of two-qubit gates might appear to be natural metrics for optimization, near-term quantum computers can also benefit from more nuanced metrics depending upon the physical system. For example, it is now established that the noise in the *IBM Q Experience* computers varies widely among qubits and also with time, and that circuit optimizations might have to be done in regular time intervals in order to exploit the current noise characteristics of the hardware [31]. In such a scenario, if we need to implement a specific logical operator at the current time, and if it is the case that some specific qubits or qubit-links in the system are particularly unreliable, then it might be better to sacrifice depth and identify an equivalent logical operator that avoids those qubits or qubit-links (if possible). As an example, for the well-known $[\![4, 2, 2]\!]$ code [6], [27], whose stabilizer group is generated as $S = \langle X_1 X_2 X_3 X_4, Z_1 Z_2 Z_3 Z_4 \rangle$, two implementations of the logical controlled-Z ($\overline{CZ}_{12}$) operation

---
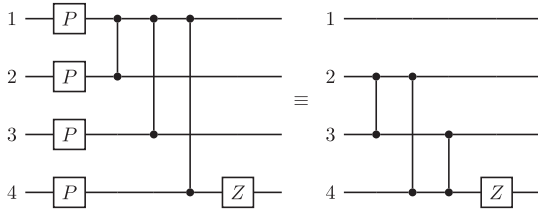
FIGURE 1. Two physical circuits that realize the CZ gate on the two logical qubits of the $[\![4, 2, 2]\!]$ code.

on the two logical qubits are shown in Fig. 1. The logical Pauli operators in this case are $\bar{X}_1 = X_1 X_2$, $\bar{X}_2 = X_1 X_3$, $\bar{Z}_1 = Z_2 Z_4$, $\bar{Z}_2 = Z_3 Z_4$.

Assuming that single-qubit gates do not contribute to complexity (or difficulty of implementation), we observe that both choices have the same number of two-qubit gates and depth. More interestingly, we see that the second choice completely avoids the first physical qubit while realizing the same logical CZ operation. Therefore, if either the first qubit itself has poor fidelity or coupling to it does, then clearly the second choice is more appropriate. Preliminary experiments on the IBM system confirm this advantage when qubits are mapped appropriately. Note that even if we use a QECC that protects a single qubit but has a transversal CZ implementation, i.e., the logical CZ is a CZ between corresponding physical qubits in two *separate* code blocks, this incurs a larger overhead than the above scheme. We identified this example by using our open-source implementation of our LCS algorithm, which is available on github.[3] In order to identify (or construct) more interesting codes that exhibit a "rich" set of choices for each logical operator, one needs a better understanding of the geometry of the space of symplectic solutions. We believe this is an important open problem arising from this article.

For near-term Noisy Intermediate-Scale Quantum (NISQ) [32] era of quantum computers, a lot of current research is focused on equipping compilers with routines that optimize circuits for depth and two-qubit gates, and the mapping of qubits from the algorithm to the hardware, while all taking into account the specific characteristics and noise in the hardware [31], [33]–[36]. Although employing QECCs is considered to be beyond the NISQ regime, exploiting simple codes, such as the $[\![4, 2, 2]\!]$ code and using post-selection provides increased reliability than uncoded computation (as Harper and Flammia have demonstrated [37], [38]). Therefore, our efficient LCS algorithm might find an application in such quantum compilers, where the utility is to determine the best physical realization of a logical operator with respect to current system characteristics. Specifically, this allows *dynamic compilation* (i.e., during program execution) that could provide significant reliability gains in practice.

[3][Online]. Available: https://github.com/nrenga/symplectic-arxiv18a

In light of such applications, our software currently allows one to determine only one physical realization in cases where the number of solutions is prohibitively large, specifically for QECCs with large-dimension stabilizers ($r = m - k \gg 1$). However, this single solution does not come with any explicit guarantees regarding depth or number of two-qubit gates or avoiding certain physical qubits. Therefore, even developing heuristics to directly optimize for a "good enough" solution, instead of assembling all solutions and searching over them, will have a significant impact on the efficiency of compilers.

B. QECCS FOR UNIVERSAL QUANTUM COMPUTATION

Physical single-qubit rotation gates on trapped-ion qubits are natural, reliable and have a long history [39]. Recently, it has also been observed that small-angle Mølmer-Sørensen gates, i.e., $XX_{ij}(\theta) = \cos\frac{\theta}{2} \cdot I_4 - \iota \sin\frac{\theta}{2} \cdot X_i X_j$ for small θ, are more reliable than the maximally-entangling $XX_{ij}(\frac{\pi}{2})$ gate [40]. Since these are the primitive operations in trapped-ion systems [41], codes that support a transversal $T = \mathrm{diag}(1, \exp(\frac{\iota\pi}{4}))$ gate, such as the tri-orthogonal codes mentioned earlier, could be directly used for computation rather than being dedicated for expensive magic state distillation [10], [42]–[44]. However, it is well-known that there exists no single QECC that supports a universal set of gates where all of them have a transversal implementation at the logical level [45]–[47]. Therefore, there is a natural tradeoff between exploiting transversality for logical non-Clifford operations versus Clifford operations.

Indeed, this will be a realistic alternative only if the logical Clifford operations on these codes are "error-resilient", by which we mean that for at least constant-depth circuits, the most likely errors remain correctable and do not propagate catastrophically through the Clifford sections of these logical circuits. For this purpose, our LCS algorithm can be a supportive tool to investigate properties of stabilizer QECCs that guarantee error-resilience of their logical Clifford operators. Note that constant-depth circuits have been shown to provide a quantum advantage over classical computation [48]. In fact, it has been shown that the advantage persists even if those circuits are noisy [49], and the proof involves a QECC which admits constant-depth logical Cliffords.

C. ORGANIZATION

The rest of this article is organized as follows. Section II discusses the connection between quantum computation and the binary symplectic group, which forms the foundation for this article. Section III begins by outlining the process of finding logical Clifford gates through a demonstration for the $[\![6, 4, 2]\!]$ CSS code [27], [50]. Then the general case of stabilizer codes is discussed rigorously via four theorems and our LCS algorithm. Finally, Section IV concludes this article. Appendix B discusses the proof of Theorem 1, and Appendix C provides the source code for Algorithm 2 with extensive comments.

II. PHYSICAL AND LOGICAL OPERATORS

Quantum error-correcting codes (QECCs) protect qubits involved in quantum computation. In this section, we summarize the mathematical framework introduced in [3], [5], [6], and [50] and described in more detail in [16] and [51]. Mathematically, an m-qubit system is treated as a Hilbert space with dimension $N = 2^m$. Universal quantum computation requires the ability to implement (within a specified tolerance) quantum operations represented by the group of $N \times N$ unitary matrices acting on this space. In this article, we are primarily concerned with the unitary operators in the Clifford group.

Notation: Let $\mathbb{R}$ denote the field of real numbers, $\mathbb{C}$ denote the field of complex numbers, and $\mathbb{F}_2$ denote the binary field. We will consider vectors over $\mathbb{F}_2$ to be row vectors and vectors over $\mathbb{R}$ or $\mathbb{C}$ to be column vectors. Vectors over $\mathbb{C}^N$ with $N = 2^m$ will be indexed by elements of $\mathbb{F}_2^m$ in the natural binary order $00\cdots00, 00\cdots01, \ldots, 11\cdots11$ (rather than $\{1, 2, \ldots, N\}$). Thus, for $v \in \mathbb{F}_2^m$, let $e_v \in \mathbb{C}^N$ denote the standard basis vector associated with index v, i.e., $e_v = |v\rangle$ is all-zero except for a 1 in the entry indexed by v.

A. PAULI MATRICES AND THE SYMPLECTIC INNER PRODUCT

For a single qubit, we have $m = 1$ and a quantum pure state is a vector in the $N = 2$ dimensional Hilbert space $\mathbb{C}^2$. A pure quantum state $\hat{u} \in \mathbb{C}^2$ is a unit-length superposition of the two states $e_0 \triangleq [1, 0]^T$, $e_1 \triangleq [0, 1]^T$ that form the *computational basis*. Thus, $\hat{u} = \alpha e_0 + \beta e_1$, where $\alpha, \beta \in \mathbb{C}$ satisfy $|\alpha|^2 + |\beta^2| = 1$. The *Pauli matrices* for a single qubit system are the 2×2 identity matrix I_2

$$X \triangleq \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \; Z \triangleq \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \; Y \triangleq \iota XZ = \begin{bmatrix} 0 & -\iota \\ \iota & 0 \end{bmatrix}. \tag{1}$$

We note that the Pauli matrices form a basis over $\mathbb{C}$ for all 2×2 complex matrices. Thus, any single qubit unitary operator (such as an error) can be written as a linear combination of Pauli matrices. One can also express any pure quantum state as $\hat{u} = (\alpha_0 I_2 + \alpha_1 X + \iota \alpha_2 Z + \alpha_3 Y) e_0$, where $\alpha_i \in \mathbb{R}$.

For an m-qubit system, we work in the $N = 2^m$ dimensional Hilbert space $\mathbb{C}^N$ and a pure quantum state $\hat{u}$ is a unit-length vector in this space. The computational basis vectors $\{e_v \in \mathbb{C}^N | (v_1, \ldots, v_m) \in \mathbb{F}_2^m\}$ are defined by the Kronecker product $e_v \triangleq e_{v_1} \otimes e_{v_2} \otimes \cdots \otimes e_{v_m}$. Thus a pure quantum state can be written as $\hat{u} = \sum_{v \in \mathbb{F}_2^m} \alpha_v e_v$, where $\sum_{v \in \mathbb{F}_2^m} |\alpha_v|^2 = 1$.

Let $N \triangleq 2^m$. Given row vectors $a, b \in \mathbb{F}_2^m$, define the m-fold Kronecker product

$$D(a, b) \triangleq X^{a_1} Z^{b_1} \otimes \cdots \otimes X^{a_m} Z^{b_m} \in \mathbb{U}_N, \tag{2}$$

where $\mathbb{U}_N$ denotes the group of all $N \times N$ unitary operators. The *Heisenberg-Weyl group* HW_N (also called the m-qubit Pauli group) consists of all operators $\iota^\kappa D(a, b)$, where

$\kappa \in \mathbb{Z}_4 \triangleq \{0, 1, 2, 3\}$. The order is $|HW_N| = 4N^2$ and the *center* of this group is $\langle \iota I_N \rangle = \{I_N, \iota I_N, -I_N, -\iota I_N\}$, where I_N is the $N \times N$ identity matrix. Since $XZ = -ZX$

$$D(a, b)D(a', b') = (-1)^{a'b^T + b'a^T} D(a', b')D(a, b) \tag{3}$$

$$= (-1)^{a'b^T} D(a + a', b + b') \tag{4}$$

and $D(a, b)^T = (-1)^{ab^T} D(a, b)$. The *symplectic inner product* in $\mathbb{F}_2^{2m}$ is defined as

$$\langle [a, b], [a', b'] \rangle_s \triangleq a'b^T + b'a^T = [a, b]\,\Omega\,[a', b']^T \pmod{2},$$

$$\text{where} \quad \Omega \triangleq \begin{bmatrix} 0 & I_m \\ I_m & 0 \end{bmatrix}. \tag{5}$$

Two operators $D(a, b)$ and $D(a', b')$ commute if and only if $\langle [a, b], [a', b'] \rangle_s = 0$. The homomorphism $\gamma : HW_N \to \mathbb{F}_2^{2m}$ defined by $\gamma(\iota^\kappa D(a, b)) \triangleq [a, b]$ for all $\kappa \in \mathbb{Z}_4$ has kernel $\langle \iota I_N \rangle$, which allows us to represent elements of HW_N (up to multiplication by scalars) as binary vectors. Since $Y = \iota XZ$ is Hermitian but XZ is not, an additional factor of ι is required to make $D(a, b)$ Hermitian for each $i \in \{1, \ldots, m\}$ where $a_i b_i = 1$. Hence, the matrix

$$E(a, b) \triangleq \iota^{ab^T \bmod 4} D(a, b) \tag{6}$$

is Hermitian and $E(a, b)^2 = I_N$, because $X^2 = Z^2 = Y^2 = I_2$. Given $[a, b], [a', b'] \in \mathbb{F}_2^{2m}$, it can be shown that

$$E(a, b)E(a', b') = (-1)^{a'b^T + b'a^T} E(a', b')E(a, b) \tag{7}$$

$$= \iota^{a'b^T - b'a^T} E(a + a', b + b') \tag{8}$$

where the exponent and the sums $a + a'$, $b + b'$ are computed modulo 4 (see [52] for the extended definition of $E(a, b)$).

B. STABILIZER CODES

We use commutative subgroups of HW_N to define resolutions of the identity. A *stabilizer* group is a commutative subgroup S of HW_N generated by commuting Hermitian matrices $\pm E(a, b)$, with the additional property that if $E(a, b) \in S$ then $-E(a, b) \notin S$. Recall that an operator is an orthogonal projection onto its range iff it is idempotent and Hermitian. Since $E(a, b)^2 = I_N$ for all $a, b \in \mathbb{F}_2^m$, the operator $\frac{I_N \pm E(a,b)}{2}$ is an orthogonal projection onto the ± 1 eigenspace of $E(a, b)$, respectively. Also, the eigenvalues of each $E(a, b)$ are ± 1 with algebraic multiplicity $N/2$.

Since all elements of S are commuting Hermitian unitary matrices, they can be simultaneously diagonalized with respect to a common orthonormal basis. We refer to such a basis as the *common eigenbasis* or simply the *eigenbasis* of the subgroup S. In addition, if the subgroup S is generated by $E(a_i, b_i), i = 1, \ldots, r$, then the operator $\frac{1}{2^r} \prod_{i=1}^r (I_N + E(a_i, b_i))$ is an orthogonal projection onto the 2^k-dimensional subspace $V(S)$ fixed pointwise by S, i.e., the $+1$ eigenspace of S, where $k \triangleq m - r$. Mathematically, $V(S) \triangleq \{|\psi\rangle \in \mathbb{C}^N | g|\psi\rangle = |\psi\rangle$ for all $g \in S\}$. The

subspace $V(S)$ is called the $[\![m, k]\!]$ *stabilizer code* determined by S, where the notation $[\![m, k]\!]$ indicates that $V(S)$ encodes k *logical* qubits into m *physical* qubits. The extended notation $[\![m, k, d]\!]$ is used to denote that any undetectable error on the code must act on at least d qubits, i.e., d is the *(minimum) distance* of the stabilizer code.

Let $\gamma(S)$ denote the subspace of $\mathbb{F}_2^{2m}$ formed by the binary representations of the elements of S under the homomorphism γ. A generator matrix for $\gamma(S)$ is $G_S \triangleq [a_i, b_i]_{i=1,\ldots,r}$ and we have $G_S \Omega G_S^T = 0_r$, where 0_r is the $r \times r$ all-zero matrix (the subscript is often neglected). The condition $G_S \Omega G_S^T = 0$ encodes the fact that elements of S must pairwise commute.

Given a stabilizer S with generators $E(a_i, b_i)$, $i = 1, \ldots, r$, we can define 2^r subgroups $S_{\epsilon_1 \cdots \epsilon_r}$ where the index $(\epsilon_1 \cdots \epsilon_r)$ represents that $S_{\epsilon_1 \cdots \epsilon_r}$ is generated by $\epsilon_i E(a_i, b_i)$, $\epsilon_i \in \{\pm 1\}$. Note that

$$\Pi_{\epsilon_1 \cdots \epsilon_r} \triangleq \frac{1}{2^r} \prod_{i=1}^{r} (I_N + \epsilon_i E(a_i, b_i)) \tag{9}$$

is the orthogonal projector onto $V(S_{\epsilon_1 \cdots \epsilon_r})$ and the sum $\sum_{(\epsilon_1, \ldots, \epsilon_r) \in \{\pm 1\}^r} \Pi_{\epsilon_1 \cdots \epsilon_r} = I_N$ defines a resolution of the identity. In quantum error correction, it is sufficient to correct Pauli errors (i.e., elements in HW_N) because general errors can be approximated by linear combinations of them [53]. Also, the elements of HW_N, acting via conjugation, permute the subgroups $S_{\epsilon_1 \cdots \epsilon_r}$. Given an $[\![m, k]\!]$ stabilizer code, it is possible to perform encoded quantum computation in any of the subspaces $V(S_{\epsilon_1 \cdots \epsilon_r})$ by synthesizing appropriate logical operators. If we think of these subspaces as *threads*, then a computation starts in one thread and jumps to another when an error (from HW_N) occurs. QECCs enable error control by identifying the jump that the computation has made. Identification makes it possible to adjust future operations in the computation instead of returning to the initial subspace and restarting the computation. The idea of tracing these threads is called *Pauli frame tracking* [54].

C. THE CLIFFORD GROUP AND SYMPLECTIC MATRICES

The Clifford group Cliff_N consists of all unitary matrices $g \in \mathbb{C}^{N \times N}$ for which $g D(a, b) g^\dagger \in HW_N$ for all $D(a, b) \in HW_N$, where $g^\dagger$ is the conjugate transpose of g [16]. Cliff_N is the *normalizer* of HW_N in the unitary group $\mathbb{U}_N$, so it contains HW_N. Note that by definition Cliff_N has an infinite center consisting of $\mathbb{U}(1) \triangleq \{e^{\iota\theta} I_N; \theta \in \mathbb{R}\}$, but it can be made finite by first taking the quotient group $\text{Cliff}_N / \mathbb{U}(1)$ and then including multiples of just the phase $e^{\iota\pi/4}$, which contributes a factor 8 in its size [5], i.e., $|\text{Cliff}_N| = 8 \cdot 2^{m^2+2m} \prod_{j=1}^{m} (4^j - 1)$. We regard operators in Cliff_N as *physical* operators acting on quantum states in $\mathbb{C}^N$, to be implemented by quantum circuits. Every operator $g \in \text{Cliff}_N$ induces an automorphism of HW_N by conjugation. Note that the inner automorphisms induced by matrices in HW_N preserve every conjugacy class $\{\pm D(a, b)\}$ and $\{\pm \iota D(a, b)\}$,

because (3) implies that elements in HW_N either commute or anti-commute. The automorphism induced by an element $g \in \text{Cliff}_N$ satisfies (see [51] for a proof)

$$g E(a, b) g^\dagger = \pm E([a, b] F_g), \quad \text{where } F_g = \begin{bmatrix} A_g & B_g \\ C_g & D_g \end{bmatrix}. \tag{10}$$

Since conjugation by g respects commutativity in HW_N, the matrix F_g preserves symplectic inner products: $\langle [a, b] F_g, [a', b'] F_g \rangle_s = \langle [a, b], [a', b'] \rangle_s$. This implies that F_g satisfies $F_g \Omega F_g^T = \Omega$. We say that F_g is a *binary symplectic matrix*, and express the symplectic property $F_g \Omega F_g^T = \Omega$ as $A_g B_g^T = B_g A_g^T$, $C_g D_g^T = D_g C_g^T$, $A_g D_g^T + B_g C_g^T = I_m$. Let $\text{Sp}(2m, \mathbb{F}_2)$ denote the group of symplectic $2m \times 2m$ matrices over $\mathbb{F}_2$. The homomorphism $\phi : \text{Cliff}_N \to \text{Sp}(2m, \mathbb{F}_2)$ defined by $\phi(g) \triangleq F_g$ is surjective with kernel $\langle HW_N, \mathbb{U}(1) \rangle$, and every Clifford operator maps down to a symplectic matrix F_g. Thus, HW_N is a normal subgroup of Cliff_N and $\text{Cliff}_N / \langle HW_N, \mathbb{U}(1) \rangle \cong \text{Sp}(2m, \mathbb{F}_2)$. This implies that the size is $|\text{Sp}(2m, \mathbb{F}_2)| = 2^{m^2} \prod_{j=1}^{m} (4^j - 1)$ (also see [5]). Table I lists elementary symplectic transformations F_g, that generate the binary symplectic group $\text{Sp}(2m, \mathbb{F}_2)$, and the corresponding unitary automorphisms $g \in \text{Cliff}_N$, which together with HW_N generate Cliff_N (see [51, Appendix I]). Some important circuit identities involving these operators are listed in [51].

In [55], Can has developed an algorithm that factors a $2m \times 2m$ binary symplectic matrix into a product of at most 6 elementary symplectic matrices of the type shown in Table I. The target symplectic matrix maps the dual basis $X_N \triangleq \{E(a, 0) : a \in \mathbb{F}_2^m\}$, $Z_N \triangleq \{E(0, b) : b \in \mathbb{F}_2^m\}$ to a dual basis X_N', Z_N'. Row and column operations by the elementary matrices return X_N', Z_N' to the original pair X_N, Z_N. This decomposition simplifies the translation of symplectic matrices into circuits (see [51, Appendix I]), and so we use it in our LCS algorithm. For completeness, we include the theorem here.

Theorem 1 ([55, Theorem 3.2.1]): Any binary symplectic transformation F can be expressed as

$$F = A_{Q_1} \Omega \, T_{R_1} G_k T_{R_2} A_{Q_2}$$

as per the notation used in Table I, where invertible matrices Q_1, Q_2 and symmetric matrices R_1, R_2 are chosen appropriately.

Proof: The idea is to perform row and column operations on the matrix F via left and right multiplication by elementary symplectic transformations from Table I, and bring the matrix F to the standard form $\Omega T_{R_1} \Omega$ (for details see Appendix B). ∎

A closely related algorithm was given earlier by Dehaene and De Moor [56]. The elementary symplectic matrices appearing in the product can be related to the Bruhat decomposition of the symplectic group (see [20]). When the algorithm is run in reverse it produces a random Clifford matrix, which

TABLE I. A universal set of logical operators for $\mathrm{Sp}(2m, \mathbb{F}_2)$ and their corresponding physical operators in Cliff_N (see Appendix A for a detailed discussion and circuits). The number of 1s in Q and R directly relates to the number of gates. Here H_{2^t} denotes the Walsh-Hadamard matrix of size 2^t, $U_t = \mathrm{diag}(I_t, 0_{m-t})$, and $L_{m-t} = \mathrm{diag}(0_t, I_{m-t})$.

Logical Operator F_g	Physical Operator g	Circuit Element
$\Omega = \begin{bmatrix} 0 & I_m \\ I_m & 0 \end{bmatrix}$	$H_N = H^{\otimes m} = \frac{1}{(\sqrt{2})^m}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}^{\otimes m}$	Transversal Hadamard
$A_Q = \begin{bmatrix} Q & 0 \\ 0 & Q^{-T} \end{bmatrix}$	$a_Q : \lvert v \rangle \mapsto \lvert vQ \rangle,\ a_Q = \sum_{v \in \mathbb{F}_2^m} \lvert vQ \rangle \langle v \rvert$	Controlled-NOT (CNOT) Qubit Permutation
$T_R = \begin{bmatrix} I_m & R \\ 0 & I_m \end{bmatrix};\ R = R^T$	$t_R = \mathrm{diag}\left(i^{vRv^T \bmod 4}\right) = \sum_{v \in \mathbb{F}_2^m} i^{vRv^T} \lvert v \rangle \langle v \rvert$	Controlled-Z (CZ) Phase (P)
$G_t = \begin{bmatrix} L_{m-t} & U_t \\ U_t & L_{m-t} \end{bmatrix}$	$g_t = H_{2^t} \otimes I_{2^{m-t}}$	Partial Hadamards

serves as a "third-order" approximation to a random unitary matrix since the Clifford group forms a unitary 3-design [57]. This is an instance of the subgroup algorithm [58] for generating uniform random variables. The algorithm has complexity $O(m^3)$ and uses $O(m^2)$ random bits, which is order optimal given the order of the symplectic group $\mathrm{Sp}(2m, \mathbb{F}_2)$ (cf. [59]). Our algorithm is similar to that developed by Jones *et al.* [60] in that it alternates (partial) Hadamard matrices and diagonal matrices; the difference is that the unitary three-design property of the Clifford group provides randomness guarantees. This also finds application in machine learning (see [61] and references therein).

III. SYNTHESIS OF LOGICAL CLIFFORD OPERATORS FOR STABILIZER CODES

Quantum computation in the protected space of an $[[m, k]]$ quantum error-correcting code (QECC) requires the translation of logical operators on the k encoded qubits into physical operators on the m code qubits. In this section, for an $[[m, k]]$ stabilizer code, we develop an algorithm that synthesizes *all* physical Clifford realizations of a logical Clifford operator, up to equivalence classes defined by their action on input Pauli operators [which is encoded in their symplectic matrix representation, by (10)]. This algorithm makes it possible to optimize the choice of circuit with respect to a metric that is a function of the quantum hardware. We now outline the algorithm and illustrate the steps using an example where we synthesize a logical controlled-Z gate on the first two logical qubits of the $[[6, 4, 2]]$ code [6], [27]. See [1] for discussions on other operators for this code.

Input: Target Clifford circuit g on the k logical qubits, stabilizers, and logical Paulis.

Output: All Clifford circuits $\bar{g}$ on the m physical qubits that preserve the code space and implement g on the k logical qubits. *Step 1:* Translate the input into linear constraints on the symplectic matrix $F_{\bar{g}}$ representing $\bar{g}$.

The stabilizer group of the $[[6, 4, 2]]$ CSS code is $S = \langle X^{\otimes 6}, Z^{\otimes 6} \rangle = \langle E(\underline{1}, \underline{0}), E(\underline{0}, \underline{1}) \rangle$, where $\underline{1} = 111111$, $\underline{0} = 000000$. The logical Pauli operators can be calculated directly [51, Section V], or using algorithms developed by Gottesman [6] or Wilde [12]. These operators are given by $\bar{X}_j = X_1 X_{j+1} = E(\underline{e}_1 + \underline{e}_{j+1}, \underline{0})$, $\bar{Z}_j = Z_{j+1} Z_6 = E(\underline{0}, \underline{e}_{j+1} + \underline{e}_6)$, $j = 1, 2, 3, 4$, where $\underline{e}_j$ is the jth standard basis vector in $\mathbb{F}_2^6$. We now find a six-qubit circuit $\overline{\mathrm{CZ}}_{12}$ on the physical (code) qubits that (i) realizes the CZ_{12} gate on the logical qubits, i.e., $\overline{\mathrm{CZ}}_{12}$ acts on $\bar{X}_j, \bar{Z}_j$ analogous to how CZ_{12} acts on X_j, Z_j as stated explicitly in (11), and (ii) preserves the code space. Note that satisfying the mathematical condition in (i) does not already guarantee (ii). The first condition is written as the constraints

$$\overline{\mathrm{CZ}}_{12} \bar{X}_j \overline{\mathrm{CZ}}_{12}^\dagger = \begin{cases} \bar{X}_1 \bar{Z}_2 & \text{if } j = 1, \\ \bar{Z}_1 \bar{X}_2 & \text{if } j = 2, \\ \bar{X}_j & \text{if } j \neq 1, 2, \end{cases}$$

$$\overline{\mathrm{CZ}}_{12} \bar{Z}_j \overline{\mathrm{CZ}}_{12}^\dagger = \bar{Z}_j \text{ for all } j = 1, 2, 3, 4. \qquad (11)$$

The symplectic representation of Clifford elements in (10) transforms these conditions into the following linear constraints on the desired symplectic matrix $F_{\overline{\mathrm{CZ}}_{12}}$:

$$[\underline{e}_1 + \underline{e}_2, \underline{0}]F_{\overline{\mathrm{CZ}}_{12}} = [\underline{e}_1 + \underline{e}_2, \underline{e}_3 + \underline{e}_6],$$

$$[\underline{e}_1 + \underline{e}_3, \underline{0}]F_{\overline{\mathrm{CZ}}_{12}} = [\underline{e}_1 + \underline{e}_3, \underline{e}_2 + \underline{e}_6],$$

$$[\underline{e}_1 + \underline{e}_{j+1}, \underline{0}]F_{\overline{\mathrm{CZ}}_{12}} = [\underline{e}_1 + \underline{e}_{j+1}, \underline{0}], \ j = 3, 4,$$

$$[\underline{0}, \underline{e}_2 + \underline{e}_6]F_{\overline{\mathrm{CZ}}_{12}} = [\underline{0}, \underline{e}_2 + \underline{e}_6],$$

$$[\underline{0}, \underline{e}_3 + \underline{e}_6]F_{\overline{\mathrm{CZ}}_{12}} = [\underline{0}, \underline{e}_3 + \underline{e}_6],$$

$$[\underline{0}, \underline{e}_{j+1} + \underline{e}_6]F_{\overline{\mathrm{CZ}}_{12}} = [\underline{0}, \underline{e}_{j+1} + \underline{e}_6], \ j = 3, 4. \qquad (12)$$

Constraint (ii) requires that the physical circuit must normalize the stabilizer. We prove later that any such circuit can be transformed into one that commutes with each stabilizer element, while realizing the same logical operation (see Theorem 11). Note that the requirement is only to preserve the code space, i.e., $\overline{CZ}_{12}$ must commute with the code projector, but this is equivalent to normalizing the stabilizer since we restrict $\overline{CZ}_{12}$ to be a physical Clifford operator. For non-Clifford physical operators, the general approach would be similar to that considered for Z-rotations in [62]. Requiring that the circuit centralize the stabilizer yields the constraints

$$[111111, 000000]F_{\overline{CZ}_{12}} = [111111, 000000],$$

$$[000000, 111111]F_{\overline{CZ}_{12}} = [000000, 111111]. \tag{13}$$

Step 2: Find all symplectic solutions.

The symplectic constraint $F_{\overline{CZ}_{12}} \Omega F_{\overline{CZ}_{12}}^T = \Omega$ is *nonlinear*, and in the description of the generic LCS algorithm that follows this example, we show how to use transvections to find all $2^{r(r+1)/2}$ symplectic solutions. We then translate each solution into a physical Clifford circuit using the decomposition of symplectic matrices as a product of the elementary matrices listed in Table I (see Appendix B or [51] for details). For the $[\![6, 4, 2]\!]$ code there are 8 symplectic solutions. The solution with smallest depth is the elementary symplectic matrix $F_{\overline{CZ}_{12}} = T_B$, where $B_{23} = B_{32} = B_{26} = B_{62} = B_{36} = B_{63} = 1$ and $B_{ij} = 0$ elsewhere. The corresponding physical operator $\overline{CZ}_{12} = \text{diag}(\iota^{vBv^T})$ can be decomposed into $CZ_{23}CZ_{26}CZ_{36}$. *Step 3:* Identify any sign violations and find a Pauli matrix to fix the signs while leaving the logical operation undisturbed.

The operator $CZ_{23}CZ_{26}CZ_{36}$ commutes with the stabilizer $E(\underline{0}, \underline{1})$ but not with the stabilizer $E(\underline{1}, \underline{0})$. Adding the Pauli operator Z_6 fixes the sign and leaves the logical operation undisturbed. So the final circuit is $CZ_{23}CZ_{26}CZ_{36}Z_6$.

A. SYMPLECTIC TRANSVECTIONS

Definition 2: Given row vector $h \in \mathbb{F}_2^{2m}$, a symplectic transvection is a map $Z_h : \mathbb{F}_2^{2m} \to \mathbb{F}_2^{2m}$ defined by

$$Z_h(x) \triangleq x + \langle x, h \rangle_s h = xF_h, \quad \text{where } F_h \triangleq I_{2m} + \Omega h^T h \tag{14}$$

where F_h is its associated symplectic matrix [59]. A transvection does not correspond to a single elementary Clifford operator.

Fact 3 ([63, Theorem 2.10]): The symplectic group $\text{Sp}(2m, \mathbb{F}_2)$ is generated by the family of symplectic transvections.

An important result that is involved in the proof of this fact is the following theorem from [59] and [63], which we restate here for $\mathbb{F}_2^{2m}$ since we will build on this result to state and prove Theorem 5.

Theorem 4: Let $x, y \in \mathbb{F}_2^{2m}$ be two nonzero vectors. Then x can be mapped to y by a product of at most two symplectic transvections.

Proof: There are two cases: $\langle x, y \rangle_s = 1$ or 0. First assume $\langle x, y \rangle_s = 1$. Define $h \triangleq x + y$, so

$$xF_h = Z_h(x) = x + \langle x, x + y \rangle_s(x + y)$$
$$= x + (\langle x, x \rangle_s + \langle x, y \rangle_s)(x + y)$$
$$= x + (0 + 1)(x + y) = y.$$

Next assume $\langle x, y \rangle_s = 0$. Define $h_1 \triangleq w + y, h_2 \triangleq x + w$, where $w \in \mathbb{F}_2^{2m}$ is chosen such that $\langle x, w \rangle_s = \langle y, w \rangle_s = 1$. Then

$$rCl + x*xF_{h_1}F_{h_2} = Z_{h_2}(x + \langle x, w + y \rangle_s(w + y))$$
$$= (x + w + y) + \langle (x + w) + y, x + w \rangle_s$$
$$\times (x + w)$$
$$= y. \qquad \blacksquare$$

We will use the above result to propose an algorithm (Algorithm 1) which determines a symplectic matrix F that satisfies $x_i F = y_i$, $i = 1, 2, \ldots, t \leq 2m$, where x_i are linearly independent and satisfy $\langle x_i, x_j \rangle_s = \langle y_i, y_j \rangle_s$ for all $i, j \in \{1, \ldots, t\}$.

B. DESCRIPTION OF THE GENERIC LOGICAL CLIFFORD SYNTHESIS (LCS) ALGORITHM

The synthesis of logical Paulis by Gottesman [6] and by Wilde [12] exploits symplectic groups over the binary field. Building on their work we have demonstrated, using the $[\![6, 4, 2]\!]$ code as an example, that the binary symplectic group provides a systematic framework for synthesizing physical implementations of any logical operator in the logical Clifford group Cliff_{2^k} for stabilizer codes. In other words, the symplectic group provides a *control plane* where effects of Clifford operators can be analyzed efficiently. For each logical Clifford operator, one can obtain all symplectic solutions using the algorithm below.

1) Collect all the linear constraints on F, obtained from the conjugation relations of the desired Clifford operator with the stabilizer generators and logical Paulis, to obtain a system of equations $UF = V$.
2) Then vectorize both sides to get $(I_{2m} \otimes U)\text{vec}(F) = \text{vec}(V)$.
3) Perform Gaussian elimination on the augmented matrix $[(I_{2m} \otimes U), \text{vec}(V)]$. If ℓ is the number of non-pivot variables in the row-reduced echelon form, then there are 2^ℓ solutions to the linear system.
4) For each such solution, check if it satisfies $F\Omega F^T = \Omega$. If it does, then it is a feasible symplectic solution for $\bar{g}$.

Clearly, this algorithm is not very efficient since ℓ could be very large. Specifically, for codes that do not encode many logical qubits this number will be very large as the system $UF = V$ will be very under-constrained. We now state and

prove two theorems that enable us to determine all symplectic solutions for each logical Clifford operator much more efficiently.

Theorem 5: Let $x_i, y_i \in \mathbb{F}_2^{2m}$, $i = 1, 2, \ldots, t \leq 2m$ be a collection of (row) vectors such that $\langle x_i, x_j \rangle_s = \langle y_i, y_j \rangle_s$. Assume that the x_i are linearly independent. Then a solution $F \in \mathrm{Sp}(2m, \mathbb{F}_2)$ to the system of equations $x_i F = y_i$ can be obtained as the product of a sequence of at most $2t$ symplectic transvections $F_h \triangleq I_{2m} + \Omega h^T h$, where $h \in \mathbb{F}_2^{2m}$ is a row vector.

Proof: We will prove this result by induction. For $i = 1$ we can simply use Theorem 4 to find $F_1 \in \mathrm{Sp}(2m, \mathbb{F}_2)$ as follows. If $\langle x_1, y_1 \rangle_s = 1$ then $F_1 \triangleq F_{h_1}$ with $h_1 \triangleq x_1 + y_1$, or if $\langle x_1, y_1 \rangle_s = 0$ then $F_1 \triangleq F_{h_{11}} F_{h_{12}}$ with $h_{11} \triangleq w_1 + y_1$, $h_{12} \triangleq x_1 + w_1$, where w_1 is chosen such that $\langle x_1, w_1 \rangle_s = \langle y_1, w_1 \rangle_s = 1$. In any case F_1 satisfies $x_1 F_1 = y_1$. Next consider $i = 2$. Let $\tilde{x}_2 \triangleq x_2 F_1$ so that $\langle x_1, x_2 \rangle_s = \langle y_1, y_2 \rangle_s = \langle y_1, \tilde{x}_2 \rangle_s$, since F_1 is symplectic and hence preserves symplectic inner products.

Similar to Theorem 4 we have two cases: $\langle \tilde{x}_2, y_2 \rangle_s = 1$ or 0. For the former, we set $h_2 \triangleq \tilde{x}_2 + y_2$ so that we clearly have $\tilde{x}_2 F_{h_2} = Z_{h_2}(\tilde{x}_2) = y_2$ (see Section III-A for the definition of $Z_h(\cdot)$). We also observe that

$$y_1 F_{h_2} = Z_{h_2}(y_1) = y_1 + \langle y_1, \tilde{x}_2 + y_2 \rangle_s (\tilde{x}_2 + y_2)$$

$$= y_1 + (\langle y_1, y_2 \rangle_s + \langle y_1, y_2 \rangle_s)(\tilde{x}_2 + y_2)$$

$$= y_1.$$

Hence, in this case, $F_2 \triangleq F_1 F_{h_2}$ satisfies $x_1 F_2 = y_1, x_2 F_2 = y_2$. For the case $\langle \tilde{x}_2, y_2 \rangle_s = 0$ we again find a w_2 that satisfies $\langle \tilde{x}_2, w_2 \rangle_s = \langle y_2, w_2 \rangle_s = 1$ and set $h_{21} \triangleq w_2 + y_2, h_{22} \triangleq \tilde{x}_2 + w_2$. Then by Theorem 4 we clearly have $\tilde{x}_2 F_{h_{21}} F_{h_{22}} = y_2$. For y_1 we observe that

$$y_1 F_{h_{21}} F_{h_{22}}$$

$$= Z_{h_{22}}\left(Z_{h_{21}}(y_1)\right)$$

$$= Z_{h_{22}}(y_1 + \langle y_1, w_2 + y_2 \rangle_s (w_2 + y_2))$$

$$= y_1 + \langle y_1, w_2 + y_2 \rangle_s (w_2 + y_2) + (\langle y_1, \tilde{x}_2 + w_2 \rangle_s$$

$$\quad + \langle y_1, w_2 + y_2 \rangle_s \langle w_2 + y_2, \tilde{x}_2 + w_2 \rangle_s)(\tilde{x}_2 + w_2)$$

$$\overset{(a)}{=} y_1 + \langle y_1, w_2 + y_2 \rangle_s (\tilde{x}_2 + y_2)$$

$$= y_1 \text{ if and only if } \langle y_1, w_2 \rangle_s = \langle y_1, y_2 \rangle_s$$

where (a) follows from $\langle y_1, \tilde{x}_2 \rangle_s = \langle y_1, y_2 \rangle_s$, $\langle w_2 + y_2, \tilde{x}_2 + w_2 \rangle_s = 1 + 0 + 0 + 1 = 0$. Hence, we pick a w_2 such that $\langle \tilde{x}_2, w_2 \rangle_s = \langle y_2, w_2 \rangle_s = 1$ and $\langle y_1, w_2 \rangle_s = \langle y_1, y_2 \rangle_s$, and then set $F_2 \triangleq F_1 F_{h_{21}} F_{h_{22}}$. Again, for this case F_2 satisfies $x_1 F_2 = y_1, x_2 F_2 = y_2$ as well.

By induction, assume F_{i-1} satisfies $x_j F_{i-1} = y_j$ for all $j = 1, \ldots, i - 1$, where $i \geq 3$. Using the same idea as for $i = 2$ above, let $x_i F_{i-1} = \tilde{x}_i$. If $\langle \tilde{x}_i, y_i \rangle_s = 1$, we simply set $F_i \triangleq F_{i-1} F_{h_i}$, where $h_i \triangleq \tilde{x}_i + y_i$. If $\langle \tilde{x}_i, y_i \rangle_s = 0$, we find a w_i that satisfies $\langle \tilde{x}_i, w_i \rangle_s = \langle y_i, w_i \rangle_s = 1$ and $\langle y_j, w_i \rangle_s =$

Algorithm 1: Algorithm to Find $F \in \mathrm{Sp}(2m, \mathbb{F}_2)$ Satisfying a Linear System of Equations, Using Theorem 5.

Input: $x_i, y_i \in \mathbb{F}_2^{2m}$ s.t.
 $\langle x_i, x_j \rangle_s = \langle y_i, y_j \rangle_s \, \forall \, i, j \in \{1, \ldots, t\}$.
Output: $F \in \mathrm{Sp}(2m, \mathbb{F}_2)$ satisfying
 $x_i F = y_i \, \forall \, i \in \{1, \ldots, t\}$
1: **if** $\langle x_1, y_1 \rangle_s = 1$ **then**
2:      set $h_1 \triangleq x_1 + y_1$ and $F_1 \triangleq F_{h_1}$.
3: **else**
4:      $h_{11} \triangleq w_1 + y_1$, $h_{12} \triangleq x_1 + w_1$ and
         $F_1 \triangleq F_{h_{11}} F_{h_{12}}$.
5: **end if**
6: **for** $i = 2, \ldots, t$ **do**
7:      Calculate $\tilde{x}_i \triangleq x_i F_{i-1}$ and $\langle \tilde{x}_i, y_i \rangle_s$.
8:      **if** $\tilde{x}_i = y_i$ **then**
9:          Set $F_i \triangleq F_{i-1}$. **Continue**.
10:     **end if**
11:     **if** $\langle \tilde{x}_i, y_i \rangle_s = 1$ **then**
12:         Set $h_i \triangleq \tilde{x}_i + y_i$, $F_i \triangleq F_{i-1} F_{h_i}$.
13:     **else**
14:         Find a w_i s.t. $\langle \tilde{x}_i, w_i \rangle_s = \langle y_i, w_i \rangle_s = 1$ and
             $\langle y_j, w_i \rangle_s = \langle y_j, y_i \rangle_s \, \forall \, j < i$.
15:         Set $h_{i1} \triangleq w_i + y_i$, $h_{i2} \triangleq \tilde{x}_i + w_i$, $F_i \triangleq$
             $F_{i-1} F_{h_{i1}} F_{h_{i2}}$.
16:     **end if**
17: **end for**
18: **return** $F \triangleq F_t$.

$\langle y_j, y_i \rangle_s \, \forall \, j < i$. Then we define $h_{i1} \triangleq w_i + y_i$, $h_{i2} \triangleq \tilde{x}_i + w_i$ and observe

$$y_j F_{h_{i1}} F_{h_{i2}} = Z_{h_{i2}}\left(Z_{h_{i1}}(y_j)\right)$$

$$= y_j + \langle y_j, w_i + y_i \rangle_s (\tilde{x}_i + y_i)$$

$$= y_j, \text{ for } j < i.$$

Again, by Theorem 4, we clearly have $\tilde{x}_i F_{h_{i1}} F_{h_{i2}} = y_i$. Hence we set $F_i \triangleq F_{i-1} F_{h_{i1}} F_{h_{i2}}$ in this case. In both cases F_i satisfies $x_j F_i = y_j \, \forall \, j = 1, \ldots, i$. Setting $F \triangleq F_t$ completes the inductive proof and it is clear that F is the product of at most $2t$ symplectic transvections. ■

The algorithm defined implicitly by the above proof is stated explicitly in Algorithm 1.

Definition 6: A symplectic basis for $\mathbb{F}_2^{2m}$ is a set of pairs $\{(v_1, w_1), (v_2, w_2), \ldots, (v_m, w_m)\}$ such that $\langle v_i, w_j \rangle_s = \delta_{ij}$ and $\langle v_i, v_j \rangle_s = \langle w_i, w_j \rangle_s = 0$, where $\delta_{ij} = 1$ if $i = j$ and 0 if $i \neq j$.

Note that the rows of any matrix in $\mathrm{Sp}(2m, \mathbb{F}_2)$ form a symplectic basis for $\mathbb{F}_2^{2m}$. There exists a symplectic Gram-Schmidt orthogonalization procedure that can produce a symplectic basis starting from the standard basis for $\mathbb{F}_2^{2m}$ and an additional vector $v \in \mathbb{F}_2^{2m}$ (see [59]).

Now, we state our main theorem, which enables one to determine all symplectic solutions for a system of linear equations.

Theorem 7: Let $\{(u_a, v_a),\ a \in \{1, \ldots, m\}\}$ be a collection of pairs of (row) vectors that form a symplectic basis for $\mathbb{F}_2^{2m}$, where $u_a, v_a \in \mathbb{F}_2^{2m}$. Consider the system of linear equations $u_i F = u_i'$, $v_j F = v_j'$, where $i \in \mathcal{I} \subseteq \{1, \ldots, m\}$, $j \in \mathcal{J} \subseteq \{1, \ldots, m\}$ and $F \in \mathrm{Sp}(2m, \mathbb{F}_2)$. Assume that the given vectors satisfy $\langle u_{i_1}, u_{i_2} \rangle_s \langle u_{i_1}', u_{i_2}' \rangle_s = 0$, $\langle v_{j_1}, v_{j_2} \rangle_s = \langle v_{j_1}', v_{j_2}' \rangle_s = 0$, $\langle u_i, v_j \rangle_s = \langle u_i', v_j' \rangle_s = \delta_{ij}$ where $i_1, i_2 \in \mathcal{I}$, $j_1, j_2 \in \mathcal{J}$ (since symplectic transformations F must preserve symplectic inner products). Let $\alpha \triangleq |\bar{\mathcal{I}}| + |\bar{\mathcal{J}}|$, where $\bar{\mathcal{I}}, \bar{\mathcal{J}}$ denote the set complements of $\mathcal{I}, \mathcal{J}$ in $\{1, \ldots, m\}$, respectively. Then there are $2^{\alpha(\alpha+1)/2}$ solutions F to the given linear system, and they can be enumerated systematically.

Proof: By the definition of a symplectic basis (Definition 6), we have $\langle u_a, v_b \rangle_s = \delta_{ab}$ and $\langle u_a, u_b \rangle_s = \langle v_a, v_b \rangle_s = 0$, where $a, b \in \{1, \ldots, m\}$. The same definition extends to any (symplectic) subspace of $\mathbb{F}_2^{2m}$. The linear system under consideration imposes constraints only on $u_i, i \in \mathcal{I}$ and $v_j, j \in \mathcal{J}$. Let W be the subspace of $\mathbb{F}_2^{2m}$ spanned by the symplectic pairs (u_c, v_c) where $c \in \mathcal{I} \cap \mathcal{J}$ and $W^\perp$ be its orthogonal complement under the symplectic inner product, i.e., $W \triangleq \langle\{(u_c, v_c),\ c \in \mathcal{I} \cap \mathcal{J}\}\rangle$ and $W^\perp \triangleq \langle\{(u_d, v_d),\ d \in \bar{\mathcal{I}} \cup \bar{\mathcal{J}}\}\rangle$, where $\bar{\mathcal{I}}, \bar{\mathcal{J}}$ denote the set complements of $\mathcal{I}, \mathcal{J}$ in $\{1, \ldots, m\}$, respectively.

Using the result of Theorem 5, we first compute one solution F_0 for the given system of equations. In the subspace W, F_0 maps $(u_c, v_c) \mapsto (u_c', v_c')$ for all $c \in \mathcal{I} \cap \mathcal{J}$ and hence we now have $W = \langle\{(u_c', v_c'),\ c \in \mathcal{I} \cap \mathcal{J}\}\rangle$ spanned by its new basis pairs (u_c', v_c'). However in $W^\perp$, F_0 maps $(u_d, v_d) \mapsto (u_d', \tilde{v}_d')$ or $(u_d, v_d) \mapsto (\tilde{u}_d', v_d')$ or $(u_d, v_d) \mapsto (\tilde{u}_d', \tilde{v}_d')$ depending on whether $d \in \mathcal{I} \cap \bar{\mathcal{J}}$ or $d \in \bar{\mathcal{I}} \cap \mathcal{J}$ or $d \in \bar{\mathcal{I}} \cap \bar{\mathcal{J}}$, respectively ($d \notin \mathcal{I} \cap \mathcal{J}$ by definition of $W^\perp$). Note however that the subspace $W^\perp$ itself is fixed. We observe that such $\tilde{u}_d'$ and $\tilde{v}_d'$ are not specified by the given linear system and hence form only a particular choice for the new symplectic basis of $W^\perp$. These can be mapped to arbitrary choices $\tilde{u}_d$ and $\tilde{v}_d$, while fixing other u_d' and v_d', as long as the new choices still complete a symplectic basis for $W^\perp$. Hence, these form the degrees of freedom for the solution set of the given system of linear equations. The number of such "free" vectors is exactly $|\bar{\mathcal{I}}| + |\bar{\mathcal{J}}| = \alpha$. This can be verified by observing that the number of basis vectors for $W^\perp$ is $2|\bar{\mathcal{I}} \cup \bar{\mathcal{J}}|$ and making the following calculation.

Number of constrained vectors in the new basis for $W^\perp$

$$= |\mathcal{I} \setminus \mathcal{J}| + |\mathcal{J} \setminus \mathcal{I}|$$
$$= |\mathcal{I}| - |\mathcal{I} \cap \mathcal{J}| + |J| - |\mathcal{I} \cap \mathcal{J}|$$
$$= (m - |\bar{\mathcal{I}}|) + (m - |\bar{\mathcal{J}}|) - 2(m - |\bar{\mathcal{I}} \cup \bar{\mathcal{J}}|)$$
$$= 2|\bar{\mathcal{I}} \cup \bar{\mathcal{J}}| - (|\bar{\mathcal{I}}| + |\bar{\mathcal{J}}|)$$
$$= 2|\bar{\mathcal{I}} \cup \bar{\mathcal{J}}| - \alpha.$$

Let $d, d_1, d_2 \in \bar{\mathcal{I}} \cup \bar{\mathcal{J}}$ be indices of some symplectic basis vectors for $W^\perp$. Then, the constraints on free vectors

$\tilde{u}_d$ and $\tilde{v}_d$ are that $\langle \tilde{u}_{d_1}, v_{d_2}' \rangle_s = \langle u_{d_1}', \tilde{v}_{d_2} \rangle_s = \langle \tilde{u}_{d_1}, \tilde{v}_{d_2} \rangle_s = \delta_{d_1 d_2}$ and all other pairs of vectors in the new basis set for $W^\perp$ be orthogonal to each other. In the dth symplectic pair — $(\tilde{u}_d, v_d')$ or $(u_d', \tilde{v}_d)$ or $(\tilde{u}_d, \tilde{v}_d)$ — of its new symplectic basis there is at least one free vector — $\tilde{u}_d$ or $\tilde{v}_d$ or both, respectively. For the first of the α free vectors, there are $2|\bar{\mathcal{I}} \cup \bar{\mathcal{J}}| - \alpha$ symplectic inner product constraints (which are linear constraints) imposed by the $2|\bar{\mathcal{I}} \cup \bar{\mathcal{J}}| - \alpha$ constrained vectors u_d', v_d'. Since $W^\perp$ has (binary) vector space dimension $2|\bar{\mathcal{I}} \cup \bar{\mathcal{J}}|$ and each linearly independent constraint decreases the dimension by 1, this leads to 2^α possible choices for the first free vector. For the second free vector, there are $\alpha - 1$-degrees of freedom as it has an additional inner product constraint from the first free vector. This leads to $2^{\alpha-1}$ possible choices for the second free vector, and so on. Therefore, the given linear system has at least $\prod_{\ell=1}^{\alpha} 2^\ell = 2^{\alpha(\alpha+1)/2}$ symplectic solutions.

We will now argue that there cannot be more solutions. The given system of equations can be represented compactly as $UF = V$, where $U, V \in \mathbb{F}_2^{(2m-\alpha) \times 2m}$ and F is symplectic. Observe that for each valid choice of V the set of symplectic solutions is disjoint, and hence they form a partition of the binary symplectic group $\mathrm{Sp}(2m, \mathbb{F}_2)$. Therefore, it is enough to show that the product of the number of such valid matrices V and $2^{\alpha(\alpha+1)/2}$ is equal to the size of $\mathrm{Sp}(2m, \mathbb{F}_2)$. By defining $k = m - \alpha$, the number of such valid matrices V is given by

$$\left[(2^{2m} - 1) \cdot (2^{2m-1} - 2^1) \cdot (2^{2m-2} - 2^2) \cdots \right.$$
$$\left. (2^{2m-(m-1)} - 2^{m-1}) \right]$$
$$\times \left[2^{2m-m} \cdot 2^{2m-(m+1)} \cdots 2^{2m-(m+k-1)} \right]$$
$$= \left(2^0 (2^{2m} - 1) \cdot 2^1 (2^{2m-2} - 1) \cdot 2^2 (2^{2m-4} - 1) \cdots \right.$$
$$\left. 2^{m-1} (2^{(m+1)-(m-1)} - 1) \cdot 2^m \right) \cdot 2^{m-1} \cdots 2^{m-(k-1)}$$
$$= 2^{\frac{m(m+1)}{2} + (k-1)m - \frac{k(k-1)}{2}} \prod_{j=1}^{m} (4^j - 1).$$

The counting in the first line is as follows. First, we assume without loss of generality that the pairs of rows i and $(m + i)$ of V form a symplectic pair, for $i = 1, \ldots, k$, and the rows $k + 1, \ldots, m$ are orthogonal to all rows of V under the symplectic inner product. More precisely, the inner products between pairs of rows of V must be the same as those between corresponding pairs of rows of U. But, we assume that we can perform a symplectic Gram–Schmidt process on U so that the above assumption is valid. For the first row of V, we can choose any nonzero vector and there are $(2^{2m} - 1)$ of them. For the second row, we need to restrict to vectors that

are orthogonal to the first row, and we need to eliminate the subspace generated by the first row. Similarly, for the third row until the mth row, we keep restricting to the subspace of vectors orthogonal to all previous rows and eliminate the subspace generated by all previous rows. For the $(m + 1)$th row, it needs to be orthogonal to all rows starting from the second to the mth, but it needs to have symplectic inner product 1 with the first row. Hence the dimension decreases by m from $2m$, but notice that the subspace generated by the first m rows cannot have any vector that has symplectic inner product 1 with the first row. Therefore, we need not subtract this subspace and this gives the count 2^{2m-m} for the $(m + 1)$th row. A similar argument can be made for all remaining rows and this completes the argument for counting. (It is easy to verify that by substituting $k = m$ above we obtain the size of $\mathrm{Sp}(2m, \mathbb{F}_2)$ exactly.) Now we expand the exponent of 2 above to obtain $\frac{1}{2}(m^2 + 2mk - k^2 - m + k)$.

Recollect that the size of the symplectic group is $2^{m^2} \prod_{j=1}^{m}(4^j - 1)$, and we need to check that the number obtained by dividing this by $2^{\alpha(\alpha+1)/2}$ is equal to the above number, for $\alpha = m - k$. Since the product $\prod_{j=1}^{m}(4^j - 1)$ matches with the expression in the count above, we only have to check that the exponents of 2 match. Here, the exponent of 2 is given by

$$m^2 - \frac{(m - k)(m - k + 1)}{2}$$
$$= \frac{1}{2}\left(2m^2 - (m^2 - 2mk + k^2 + m - k)\right)$$
$$= \frac{1}{2}\left(m^2 + 2mk - k^2 - m + k\right),$$

which equals the exponent calculated above. This completes the proof that the given system has exactly $2^{\alpha(\alpha+1)/2}$ solutions.

Finally, we show how to get each symplectic solution F for the given linear system. First form the matrix A whose rows are the new symplectic basis vectors for $\mathbb{F}_2^{2m}$ obtained under the action of F_0, i.e., the first m rows are $u'_c, u'_d, \tilde{u}'_d$ and the last m rows are $v'_c, v'_d, \tilde{v}'_d$. Observe that this matrix is symplectic and invertible. Then form a matrix $B = A$ and replace the rows corresponding to free vectors with a particular choice of free vectors, chosen to satisfy the conditions mentioned above. Note that B and A differ in exactly α rows, and that B is also symplectic and invertible. Determine the symplectic matrix $F' = A^{-1}B$ which fixes all new basis vectors obtained for W and $W^{\perp}$ under F_0 except the free vectors in the basis for $W^{\perp}$. Then this yields a new solution $F = F_0 F'$ for the given system of linear equations. Note that if $\tilde{u}_d = \tilde{u}'_d$ and $\tilde{v}_d = \tilde{v}'_d$ for all free vectors, where $\tilde{u}'_d, \tilde{v}'_d$ were obtained under the action of F_0 on $W^{\perp}$, then $F' = I_{2m}$. Repeating this process for all $2^{\alpha(\alpha+1)/2}$ choices of free vectors enumerates all the solutions for the linear system under consideration. ∎

Remark 8: For any system of symplectic linear equations $x_i F = y_i$, $i = 1, \ldots, t$ where the x_i do not form a symplectic basis for $\mathbb{F}_2^{2m}$, we first calculate a symplectic

---

Algorithm 2: Algorithm to Determine All $F \in \mathrm{Sp}(2m, \mathbb{F}_2)$ Satisfying a Linear System of Equations, Using Theorem 7.

Input: $u_a, v_b \in \mathbb{F}_2^{2m}$ s.t. $\langle u_a, v_b \rangle_\mathrm{s} = \delta_{ab}$ and $\langle u_a, u_b \rangle_\mathrm{s} = \langle v_a, v_b \rangle_\mathrm{s} = 0$, where $a, b \in \{1, \ldots, m\}$. $u'_i, v'_j \in \mathbb{F}_2^{2m}$ s.t. $\langle u'_{i_1}, u'_{i_2} \rangle_\mathrm{s} = 0$, $\langle v'_{j_1}, v'_{j_2} \rangle_\mathrm{s} = 0$, $\langle u'_i, v'_j \rangle_\mathrm{s} = \delta_{ij}$, where $i, i_1, i_2 \in \mathcal{I}$, $j, j_1, j_2 \in \mathcal{J}$, $\mathcal{I}, \mathcal{J} \subseteq \{1, \ldots, m\}$.

Output: $\mathcal{F} \subset \mathrm{Sp}(2m, \mathbb{F}_2)$ such that each $F \in \mathcal{F}$ satisfies $u_i F = u'_i \; \forall \, i \in \mathcal{I}$, and $v_j F = v'_j \; \forall \, j \in \mathcal{J}$.

1: Determine a particular symplectic solution F_0 for the linear system using Algorithm 1.
2: Form the matrix A whose ath row is $u_a F_0$ and $(m + b)$th row is $v_b F_0$, where $a, b \in \{1, \ldots, m\}$.
3: Compute the inverse of this matrix, A^{-1}, in $\mathbb{F}_2$.
4: Set $\mathcal{F} = \phi$ and $\alpha \triangleq |\bar{\mathcal{I}}| + |\bar{\mathcal{J}}|$, where $\bar{\mathcal{I}}, \bar{\mathcal{J}}$ denote the set complements of $\mathcal{I}, \mathcal{J}$ in $\{1, \ldots, m\}$, respectively.
5: **for** $\ell = 1, \ldots, 2^{\alpha(\alpha+1)/2}$ **do**
6:      Form a matrix $B_\ell = A$.
7:      For $i \notin \mathcal{I}$ and $j \notin \mathcal{J}$ replace the ith and $(m + j)$th rows of B_ℓ with arbitrary vectors such that $B_\ell \Omega B_\ell^T = \Omega$ and $B_\ell \neq B_{\ell'}$ for $1 \leq \ell' < \ell$.
        /* See proof of Theorem 7 for details or Appendix C for example MATLAB code */
8:      Compute $F' = A^{-1}B$.
9:      Add $F_\ell \triangleq F_0 F'$ to $\mathcal{F}$.
10: **end for**
11: **return** $\mathcal{F}$

---

basis (u_j, v_j), $j = 1, \ldots, m$ using the symplectic Gram–Schmidt orthogonalization procedure discussed in [59]. Then we transform the given system into an equivalent system of constraints on these basis vectors u_j, v_j and apply Theorem 7 to obtain all symplectic solutions.

The algorithm defined implicitly by the above proof is stated explicitly in Algorithm 2. For a given system of linear (independent) equations, if $\alpha = 0$ then the symplectic matrix F is fully constrained and there is a unique solution. Otherwise, the system is partially constrained and we refer to a solution F as a *partial* symplectic matrix.

Example: As an application of this theorem, we discuss the procedure to determine all symplectic solutions for the logical controlled-Z gate $\overline{\mathrm{CZ}}_{12}$ discussed at the beginning of this section. First we define a symplectic basis for $\mathbb{F}_2^{12}$ using the binary vector representation of the logical Pauli operators and stabilizer generators of the $[\![6, 4, 2]\!]$ code.

$$u_1 \triangleq [110000, 000000], \qquad v_1 \triangleq [000000, 010001],$$
$$u_2 \triangleq [101000, 000000], \qquad v_2 \triangleq [000000, 001001],$$
$$u_3 \triangleq [100100, 000000], \qquad v_3 \triangleq [000000, 000101],$$
$$u_4 \triangleq [100010, 000000], \qquad v_4 \triangleq [000000, 000011],$$

$$u_5 \triangleq [111111, 000000], \qquad v_5 \triangleq [000000, 000001],$$

$$u_6 \triangleq [100000, 000000], \qquad v_6 \triangleq [000000, 111111].$$
$$\tag{15}$$

Note that v_5 and u_6 do not correspond to either a logical Pauli operator or a stabilizer element but were added to complete a symplectic basis. Hence we have $\mathcal{I} = \{1, 2, 3, 4, 5\}$, $\mathcal{J} = \{1, 2, 3, 4, 6\}$ and $\alpha = 1 + 1 = 2$. As discussed earlier, we impose constraints on all u_i, v_j except for $i = 6$ and $j = 5$. Therefore, as per the notation in the above proof, we have $W \triangleq \langle \{(u_1, v_1), \ldots, (u_4, v_4)\} \rangle$ and $W^{\perp} \triangleq \langle \{(u_5, v_5), (u_6, v_6)\} \rangle$. Using Algorithm 1 we obtain a particular solution $F_0 = T_B$ where B is given in the beginning of Section III. Then we compute the action of F_0 on the bases for W and $W^{\perp}$ to get

$$u_i F_0 \triangleq u_i', \ v_j F_0 \triangleq v_j', \ i \in \mathcal{I}, \ j \in \mathcal{J}, \ \text{and}$$

$$u_6 F_0 = [100000, 000000] \triangleq \tilde{u}_6',$$

$$v_5 F_0 = [000000, 000001] \triangleq \tilde{v}_5', \tag{16}$$

where u_i', v_j' are the vectors obtained in (12), (13). Then we identify $\tilde{v}_5$ and $\tilde{u}_6$ to be the free vectors and one particular solution is $\tilde{v}_5 = \tilde{v}_5'$, $\tilde{u}_6 = \tilde{u}_6'$. In this case we have $2^{\alpha} = 2^2 = 4$ choices to pick $\tilde{v}_5$, since we need $\langle u_5', \tilde{v}_5 \rangle_s = 1$, $\langle u_i', \tilde{v}_5 \rangle_s = 0$ for $i = 1, 2, 3, 4$, and $\langle v_j, \tilde{v}_5 \rangle_s = 0$ for $j = 1, 2, 3, 4, 6$. For each such choice of $\tilde{v}_5$, we have $2^{\alpha-1} = 2$ choices for $\tilde{u}_6$. Next we form the matrix A whose ith row is u_i' and $(6 + j)$th row is v_j', where $i \in \mathcal{I}, j \in \mathcal{J}$. We set the 6th row to be $\tilde{u}_6'$ and the 11th row to be $\tilde{v}_5'$. Then we form a matrix $B = A$ and replace rows 6 and 11 by one of the eight possible pair of choices for $\tilde{u}_6$ and $\tilde{v}_5$, respectively. This yields the matrix $F' = A^{-1}B$ and the symplectic solution $F = F_0 F'$. Looping through all the eight choices we obtain the solutions listed in the appendices of [51].

Theorem 9: For an $[\![m, k]\!]$ stabilizer code, the number of solutions for each logical Clifford operator is $2^{r(r+1)/2}$, ignoring stabilizer degrees of freedom (Remark 12), where $r = m - k$.

Proof: Let $u_i, v_i \in \mathbb{F}_2^{2m}$ represent the logical Pauli operators $\bar{X}_i, \bar{Z}_i$, for $i = 1, \ldots, k$, respectively, i.e., $\gamma(\bar{X}_i) = u_i, \gamma(\bar{Z}_i) = v_i$, where γ is the map defined in Section II-A. Since $\bar{X}_i \bar{Z}_i = -\bar{Z}_i \bar{X}_i$ and $\bar{X}_i \bar{Z}_j = \bar{Z}_j \bar{X}_i$ for all $j \neq i$, it is clear that $\langle u_i, v_j \rangle_s = \delta_{ij}$ for $i, j \in \{1, \ldots, k\}$ and hence they form a partial symplectic basis for $\mathbb{F}_2^{2m}$. Let $u_{k+1}, \ldots, u_m$ represent the stabilizer generators, i.e., $\gamma(S_j) = u_{k+j}$ where the stabilizer group is $S = \langle S_1, \ldots, S_r \rangle$. Since by definition $\bar{X}_i, \bar{Z}_i$ commute with all stabilizer elements, it is clear that $\langle u_i, u_j \rangle_s = \langle v_i, u_j \rangle_s = 0$ for $i \in \{1, \ldots, k\}, j \in \{k+1, \ldots, m\}$. To complete the symplectic basis we find vectors $v_{k+1}, \ldots, v_m$ s.t. $\langle u_i, v_j \rangle_s = \delta_{ij} \ \forall \ i, j \in \{1, \ldots, m\}$. Now we note that for any logical Clifford operator, the conjugation relations with logical Paulis yield $2k$ constraints, on u_i, v_i for $i \in \{1, \ldots, k\}$, and the normalization condition on the stabilizer yields r constraints, on

$u_{k+1}, \ldots, u_m$. Hence we have $\bar{\mathcal{I}} = \phi$, $\bar{\mathcal{J}} = \{k+1, \ldots, m\}$, as per the notation in Theorem 7, and thus, $\alpha = |\bar{\mathcal{I}}| + |\bar{\mathcal{J}}| = m - k = r$. ∎

Corollary 10: For any logical k-qubit Clifford operation on an $[\![m, k]\!]$ stabilizer code, there always exists a physical m-qubit Clifford circuit that normalizes the stabilizer and realizes the given operation. Effectively, this identifies the surjection $\text{Cliff}_N \cap \mathcal{N}_{\mathbb{U}_N}(S) \to \text{Cliff}_{2^k}$ whose kernel is all the physical Cliffords that normalize the stabilizer but realize only the logical identity (see the proof of Theorem 11 for a method to identify them). Here, $\mathcal{N}_{\mathbb{U}_N}(S)$ denotes the normalizer of S in the group $\mathbb{U}_N$ of all m-qubit unitary operations.

Note that, for each symplectic solution, there are multiple decompositions into elementary forms (from Table I) possible; one possibility is given in Theorem 1. Although each decomposition yields a different circuit, all of them will act identically on X_N and Z_N under conjugation (see Section II-C for notation). Once a logical Clifford operator is defined by its conjugation with the logical Pauli operators, a physical realization of the operator could either normalize the stabilizer or centralize it, i.e., fix each element of the stabilizer group under conjugation. We note here that any obtained normalizing solution can be converted into a centralizing solution. While we do not have a well-motivated application for this result yet, we believe this might be useful in Pauli frame tracking [54] and adapting future logical operations to the current signs.

Theorem 11: For an $[\![m, k]\!]$ stabilizer code with stabilizer S, each physical realization of a given logical Clifford operator that normalizes S can be converted into a circuit that centralizes S while realizing the same logical operation.

Proof: Let the symplectic solution for a specific logical Clifford operator $\bar{g} \in \text{Cliff}_N$ that normalizes the stabilizer S be denoted by F_n. Define the logical Pauli groups $\bar{X} \triangleq \langle \bar{X}_1, \ldots, \bar{X}_k \rangle$ and $\bar{Z} \triangleq \langle \bar{Z}_1, \ldots, \bar{Z}_k \rangle$. Let $\gamma(\bar{X})$ and $\gamma(\bar{Z})$ denote the matrices whose rows are $\gamma(\bar{X}_i)$ and $\gamma(\bar{Z}_i)$, respectively, for $i = 1, \ldots, k$, where γ is the map defined in Section II-A. Similarly, let $\gamma(S)$ denote the matrix whose rows are the images of the stabilizer generators under the map γ. Then, by stacking these matrices as in the proof of Theorem 9, we observe that F_n is a solution of the linear system

$$\begin{bmatrix} \gamma(\bar{X}) \\ \gamma(S) \\ \gamma(\bar{Z}) \end{bmatrix} F_n = \begin{bmatrix} \gamma(\bar{X}') \\ \gamma(S') \\ \gamma(\bar{Z}') \end{bmatrix},$$

where $\bar{X}', \bar{Z}'$ are defined by the conjugation relations of $\bar{g}$ with the logical Paulis, i.e., $\bar{g}\bar{X}_i\bar{g}^{\dagger} = \bar{X}_i'$, $\bar{g}\bar{Z}_i\bar{g}^{\dagger} = \bar{Z}_i'$, and S' denotes the stabilizer group of the code generated by a different set of generators than that of S. Note, however, that as a group $S' = S$. The goal is to find a different solution F_c that centralizes the stabilizer, i.e. we replace $\gamma(S')$ with $\gamma(S)$ above.

We first find a matrix $K \in \mathrm{GL}(m - k, \mathbb{F}_2)$ such that $K\gamma(S') = \gamma(S)$, which always exists since generators of S' span S as well. Then we determine a symplectic solution H for the linear system

$$
\begin{bmatrix} \gamma(\bar{X}) \\ \gamma(S) \\ \gamma(\bar{Z}) \end{bmatrix} H = \begin{bmatrix} \gamma(\bar{X}) \\ K\gamma(S) \\ \gamma(\bar{Z}) \end{bmatrix},
$$

so that H satisfies $K\gamma(S) = \gamma(S)H$ while fixing $\gamma(\bar{X})$ and $\gamma(\bar{Z})$. Then since K is invertible we can write

$$
\begin{bmatrix} I_k & & \\ & K & \\ & & I_k \end{bmatrix} \begin{bmatrix} \gamma(\bar{X}) \\ \gamma(S) \\ \gamma(\bar{Z}) \end{bmatrix} F_n = \begin{bmatrix} I_k & & \\ & K & \\ & & I_k \end{bmatrix} \begin{bmatrix} \gamma(\bar{X}') \\ \gamma(S') \\ \gamma(\bar{Z}') \end{bmatrix}
$$

$$
\Rightarrow \begin{bmatrix} \gamma(\bar{X}) \\ \gamma(S) \\ \gamma(\bar{Z}) \end{bmatrix} HF_n = \begin{bmatrix} \gamma(\bar{X}') \\ \gamma(S) \\ \gamma(\bar{Z}') \end{bmatrix}.
$$

Hence, $F_c \triangleq HF_n$ is a centralizing solution for $\bar{g}$. Note that there are $2^{r(r+1)/2}$ solutions for H, as per the result of Theorem 9 with the operator being the identity operator on the logical qubits, and these produce all centralizing solutions for $\bar{g}$. ∎

Although any normalizing solution can be converted into a centralizing solution, the optimal solution with respect to a suitable metric need not always centralize the stabilizer. However, we can always setup the problem of identifying a symplectic matrix, representing the physical circuit, by constraining it to centralize the stabilizer. The general procedure to determine all symplectic solutions, and their circuits, for a logical Clifford operator for a stabilizer code is summarized in Algorithm 3. For the $[[6, 4, 2]]$ CSS code, we employed Algorithm 3 to determine the solutions listed in the appendices of [51] for each of the standard generating operators for the Clifford group (see Table I).

The `MATLAB` programs for all algorithms in this paper are available at.[4] We executed our programs on a laptop running the Windows 10 operating system (64-bit) with an Intel Core i7-5500U @ 2.40GHz processor and 8GB RAM. For the $[[6, 4, 2]]$ CSS code, it takes about 0.5 seconds to generate all 8 symplectic solutions and their circuits for one logical Clifford operator. For the $[[5, 1, 3]]$ perfect code, it takes about 20 seconds to generate all 1024 solutions and their circuits. Note that for step 5 in Algorithm 3, we use 1-qubit and 2-qubit unitary matrices (from Cliff_{2^2}) to calculate conjugations for the Pauli operator on each qubit, at each circuit element at each depth, and then combine the results to compute the conjugation of $\bar{g}$ with a stabilizer generator or logical Pauli operator. Owing to our naive implementation, we observe that most of the time is consumed in computing Kronecker products and not in calculating the symplectic solutions.

[4][Online]. Available: https://github.com/nrenga/symplectic-arxiv18a

---

Algorithm 3: LCS Algorithm to Determine All Logical Clifford Operators (See Section II for the Homomorphisms γ, ϕ).

1: Determine the target logical operator $\bar{g}$ by specifying its action on logical Paulis $\bar{X}_i$, $\bar{Z}_i$ [16]: $\bar{g}\bar{X}_i\bar{g}^\dagger = \bar{X}'_i$, $\bar{g}\bar{Z}_i\bar{g}^\dagger = \bar{Z}'_i$.

2: Transform the above relations into linear equations on $F \in \mathrm{Sp}(2m, \mathbb{F}_2)$ using the map γ and the result of (10), i.e., $\gamma(\bar{X}_i)F = \gamma(\bar{X}'_i)$, $\gamma(\bar{Z}_i)F = \gamma(\bar{Z}'_i)$. Add the conditions for normalizing the stabilizer S, i.e., $\gamma(S)F = \gamma(S')$.

3: Calculate the feasible symplectic solution set $\mathcal{F}$ using Algorithm 2 by mapping $\bar{X}_i$, S, $\bar{Z}_i$ to u_i, v_i as in Theorem 9.

4: Factor each $F \in \mathcal{F}$ into a product of elementary symplectic transformations listed in Table I, possibly using the algorithm given in [64] (which is restated in Theorem 1 here), and compute the physical Clifford operator $\bar{g}$.

5: Check for conjugation of $\bar{g}$ with the stabilizer generators and for the conditions derived in step 1. If some signs are incorrect, post-multiply by an element from HW_N as necessary to satisfy all these conditions (apply [65, Prop. 10.4] for $S^\perp = \langle S, \bar{X}_i, \bar{Z}_i \rangle$, using γ). Since HW_N is the kernel of the map ϕ, post-multiplication does not change F.

6: Express $\bar{g}$ as a sequence of physical Clifford gates corresponding to the elementary symplectic matrices obtained from the factorization in step 4 (see Appendix A for the circuits for these matrices).

Remark 12: Observe that, in our LCS algorithm, we are not taking into account the degrees of freedom provided by stabilizers. That is, if the logical operator $\bar{g}$ is required to map $\bar{X}_i \mapsto \bar{X}'_i$, then an equivalent condition is to map $\bar{X}_i \mapsto \bar{X}'_i \cdot s$, where $s \in S$ is any stabilizer element for the given code. A similar statement is true for $\bar{Z}_i \mapsto \bar{Z}'_i$. An explicit example for this scenario is the $\overline{\mathrm{CNOT}}_{1\to 2}$ for the $[[4, 2, 2]]$ code with the logical Paulis defined instead as $\bar{X}_1 = X_1X_2$, $\bar{X}_2 = X_2X_4$, $\bar{Z}_1 = Z_1Z_3$, $\bar{Z}_2 = Z_3Z_4$. The operation $\overline{\mathrm{CNOT}}_{1\to 2}$ can simply be defined as swapping qubits 2 and 4, but this maps $\bar{Z}_2 \mapsto Z_2Z_3 = \bar{Z}_1\bar{Z}_2 \cdot g^Z$, where $g^Z = Z_1Z_2Z_3Z_4$, instead of just $\bar{Z}_2 \mapsto \bar{Z}_1\bar{Z}_2$ as the above algorithm would typically require.

In principle, the LCS algorithm can be easily modified to consider these possibilities, but this significantly increases the computational complexity of the algorithm. A better understanding of the structure of logical Clifford operators for a given general stabilizer code, or even heuristics developed to identify which degrees of freedom are worth considering for

a given code, would greatly improve the quality of solutions produced by the overall algorithm.

IV. CONCLUSION

In this article, we have used the binary symplectic group to propose a systematic algorithm for synthesizing physical (Clifford) implementations of logical Clifford operators for any stabilizer code. This algorithm provides as solutions all symplectic matrices corresponding to the desired logical operator, each of which is subsequently transformed into a circuit by decomposing it into elementary forms. This decomposition is not unique, and in future work we will address optimization of the synthesis algorithm with respect to circuit complexity, error-resilience, and also other nuanced metrics discussed in the introduction. For such optimization to be feasible, one might have to explore opportunities for identifying and exploring the algebraic structure hidden in the algorithm, since combinatorially the matrix inversion involved in Algorithm 2 could itself form a bottleneck.

APPENDIX A: ELEMENTARY SYMPLECTIC TRANSFORMATIONS AND THEIR CIRCUITS

In this section, we verify that the physical operators listed in Table I are associated with the corresponding symplectic transformation [64]. Furthermore, we also provide circuits that realize these physical operators (also see [56]).

Since each physical operator in Table I is a unitary Clifford operator, it is enough to consider their actions on elements of the Heisenberg-Weyl group HW_N, where $N = 2^m$. Let e_v be a standard basis (column) vector in $\mathbb{C}^N$ indexed by the vector $v \in \mathbb{F}_2^m$ such that it has entry 1 in position v and 0 elsewhere. More precisely, if $v = [v_1, v_2, \ldots, v_m]$ then $e_v = e_{v_1} \otimes e_{v_2} \otimes \cdots e_{v_m}$, where $e_0 \triangleq \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle, e_1 \triangleq \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle$. Hence, we can simply write $e_v = |v\rangle = |v_1\rangle \otimes \cdots \otimes |v_m\rangle$.

1) $H_N = H^{\otimes m}$: The single-qubit Hadamard operator $H \triangleq \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$ satisfies $HXH^\dagger = Z, HZH^\dagger = X$. Hence, the action of H_N on a HW_N element $D(a, b)$ is given by

$$H_N D(a, b) H_N^\dagger = H_N D(a, 0) D(0, b) H_N^\dagger \tag{17}$$

$$= (H_N D(a, 0) H_N^\dagger)$$

$$(H_N D(0, b) H_N^\dagger) \tag{18}$$

$$= D(0, a) D(b, 0) \tag{19}$$

$$= (-1)^{ab^T} D(b, a) \tag{20}$$

$$\Rightarrow H_N D(a, b) H_N^\dagger = (-1)^{ab^T} D([a, b]\Omega). \tag{21}$$

The circuit for H_N is just H applied to each of the m qubits.

2) $GL(m, \mathbb{F}_2)$: Each nonsingular $m \times m$ binary matrix Q is associated with a symplectic transformation A_Q given by

$$A_Q = \begin{bmatrix} Q & 0 \\ 0 & Q^{-T} \end{bmatrix}, \tag{22}$$

where $Q^{-T} = (Q^T)^{-1} = (Q^{-1})^T$. The matrix Q is also associated with the unitary operator a_Q which realizes the mapping $e_v \mapsto e_{vQ}$. We verify this as follows. Note that $D(c, 0)e_v = e_{v+c}$ and $D(0, d)e_v = (-1)^{vd^T} e_v$. We calculate $(a_Q D(c, d) a_Q^\dagger)e_v$

$$= a_Q D(c, 0) D(0, d) e_{vQ^{-1}} \tag{23}$$

$$= a_Q (-1)^{cd^T} D(0, d) D(c, 0) e_{vQ^{-1}} \tag{24}$$

$$= (-1)^{cd^T} a_Q (-1)^{(vQ^{-1}+c)d^T} e_{vQ^{-1}+c} \tag{25}$$

$$= (-1)^{cd^T} (-1)^{(v+cQ)Q^{-1}d^T} e_{v+cQ} \tag{26}$$

$$= (-1)^{cd^T} D(0, d(Q^{-1})^T) D(cQ, 0) e_v \tag{27}$$

$$= D(cQ, dQ^{-T}) e_v \tag{28}$$

$$= D([c, d]A_Q) e_v. \tag{29}$$

Since the operator a_Q realizes the map $e_v = |v\rangle \mapsto |vQ\rangle$, the circuit for the operator is equivalent to the binary circuit that realizes $v \mapsto vQ$. Evidently, this elementary transformation encompasses CNOT operations and qubit permutations. For the latter, Q will be a permutation matrix. Note that if a_Q preserves the code space of a CSS code then the respective permutation must be in the automorphism group of the constituent classical code. This is the special case that is discussed in detail by Grassl and Roetteler in [25].

For a general Q, one can use the LU decomposition over $\mathbb{F}_2$ to obtain $P_\pi Q = LU$, where P_π is a permutation matrix, L is lower triangular and U is upper triangular. Note that $L_{ii} = U_{ii} = 1 \ \forall \ i \in \{1, \ldots, m\}$. Then the circuit for Q first involves the permutation P_π^T (or π^{-1}), then CNOTs for L with control qubits in the order $1, 2, \ldots, m$ and then CNOTs for U with control qubits in reverse order $m, m-1, \ldots, 1$. The order is important because an entry $L_{ji} = 1$ implies a CNOT gate with qubit j controlling qubit i (with $j > i$), i.e, $\text{CNOT}_{j \to i}$, and similarly $L_{kj} = 1$ implies the gate $\text{CNOT}_{k \to j}$ (with $k > j$). Since the gate $\text{CNOT}_{j \to i}$ requires the value of qubit j *before* it is altered by $\text{CNOT}_{k \to j}$, it needs to be implemented first. A similar reasoning applies to the reverse order of control qubits for U.

3) $t_R = \text{diag}(\imath^{vRv^T})$: Each symmetric matrix $R \in \mathbb{F}_2^{m \times m}$ is associated with a symplectic transformation T_R given

by

$$T_R = \begin{bmatrix} I_m & R \\ 0 & I_m \end{bmatrix}, \tag{30}$$

and with a unitary operator t_R that realizes the map $e_v \mapsto \imath^{vRv^T} e_v$. We now verify that conjugation by t_R induces T_R. We calculate $(t_R D(a, b) t_R^\dagger) e_v$

$$= \imath^{-vRv^T} t_R (-1)^{ab^T} D(0, b) D(a, 0) e_v \tag{31}$$

$$= \imath^{-vRv^T} (-1)^{ab^T} t_R (-1)^{(v+a)b^T} e_{v+a} \tag{32}$$

$$= (-1)^{ab^T} \imath^{-vRv^T} (-1)^{(v+a)b^T} \imath^{(v+a)R(v+a)^T} e_{v+a} \tag{33}$$

$$= (-1)^{ab^T} \imath^{aRa^T} (-1)^{vRa^T + (v+a)b^T} e_{v+a} \tag{34}$$

$$= (-1)^{ab^T} \imath^{-aRa^T} (-1)^{(v+a)(b+aR)^T} e_{v+a} \tag{35}$$

$$= (-1)^{ab^T} \imath^{-aRa^T} D(0, b + aR) D(a, 0) e_v \tag{36}$$

$$= (-1)^{ab^T} \imath^{-aRa^T} (-1)^{a(b+aR)^T} D(a, b + aR) e_v \tag{37}$$

$$= \imath^{aRa^T} D([a, b] T_R) e_v. \tag{38}$$

Hence, for $E(a, b) \triangleq \imath^{ab^T} D(a, b)$, we have $t_R E(a, b) t_R^\dagger = \imath^{ab^T} \imath^{aRa^T} D(a, b + aR) = E([a, b] T_R)$ as required. We derive the circuit for this unitary operator by observing the action of T_R on the standard basis vectors $[\underline{e}_1, \underline{0}], \dots, [\underline{e}_m, \underline{0}], [\underline{0}, \underline{e}_1], \dots, [\underline{0}, \underline{e}_m]$ of $\mathbb{F}_2^{2m}$, where $i \in \{1, \dots, m\}$, which captures the effect of t_R on the (basis) elements $X_1, \dots, X_m$, $Z_1, \dots, Z_m$ of HW_N, respectively, under conjugation. Assume as the first special case that R has nonzero entries only in its (main) diagonal. If $R_{ii} = 1$ then we have $[\underline{e}_i, \underline{0}] T_R = [\underline{e}_i, \underline{e}_i]$. This indicates that t_R maps $X_i \mapsto X_i Z_i \approx Y_i$. Since we know that the phase gate P_i on the ith qubit performs exactly this map under conjugation, we conclude that the circuit for t_R involves P_i. We proceed similarly for every $i \in \{1, \dots, m\}$ such that $R_{ii} = 1$.

Now consider the case where $R_{ij} = R_{ji} = 1$ (since R is symmetric). Then, we have

$$[\underline{e}_i, \underline{0}] T_R = [\underline{e}_i, \underline{e}_j], \quad [\underline{e}_j, \underline{0}] T_R = [\underline{e}_j, \underline{e}_i]. \tag{39}$$

This indicates that t_R maps $X_i \mapsto X_i Z_j$ and $X_j \mapsto Z_i X_j$. Since we know that the controlled-Z gate CZ_{ij} on qubits (i, j) performs exactly this map under conjugation, we conclude that the circuit for t_R involves CZ_{ij}. We proceed similarly for every pair (i, j) such that $R_{ij} = R_{ji} = 1$.

Finally, we note that the symplectic transformation associated with the operator $H_N t_R H_N$ is $\Omega T_R \Omega = \begin{bmatrix} I_m & 0 \\ R & I_m \end{bmatrix}$.

4) $g_t = H_{2^t} \otimes I_{2^{m-t}}$: Since H_{2^t} is the t-fold Kronecker product of H and since $D(a, b) = X^{a_1} Z^{b_1} \otimes \cdots \otimes$

$X^{a_m} Z^{b_m}$, we have $g_t D(a, b) g_t^\dagger$

$$= (Z^{a_1} X^{b_1} \otimes \cdots \otimes Z^{a_t} X^{b_t})$$

$$\otimes (X^{a_{t+1}} Z^{b_{t+1}} \otimes \cdots \otimes X^{a_m} Z^{b_m}) \tag{40}$$

$$= ((-1)^{a_1 b_1} X^{b_1} Z^{a_1} \otimes \cdots \otimes (-1)^{a_t b_t} X^{b_t} Z^{a_t})$$

$$\otimes (X^{a_{t+1}} Z^{b_{t+1}} \otimes \cdots \otimes X^{a_m} Z^{b_m}). \tag{41}$$

We write $(a, b) = (\hat{a}\bar{a}, \hat{b}\bar{b})$, where $\hat{a} \triangleq a_1 \cdots a_t$, $\bar{a} \triangleq a_{t+1} \cdots a_m$, $\hat{b} \triangleq b_1 \cdots b_t$, $\bar{b} \triangleq b_{t+1} \cdots b_m$. Then

$$g_t D(\hat{a}\bar{a}, \hat{b}\bar{b}) g_t^\dagger = (-1)^{\hat{a}\hat{b}^T} D(\hat{b}\bar{a}, \hat{a}\bar{b}) \tag{42}$$

$$= (-1)^{\hat{a}\hat{b}^T} D([\hat{a}\bar{a}, \hat{b}\bar{b}] G_t), \tag{43}$$

$$\text{where } G_t = \begin{bmatrix} 0 & 0 & I_t & 0 \\ 0 & I_{m-t} & 0 & 0 \\ I_t & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{m-t} \end{bmatrix}. \tag{44}$$

Defining $U_t \triangleq \begin{bmatrix} I_t & 0 \\ 0 & 0 \end{bmatrix}$, $L_{m-t} \triangleq \begin{bmatrix} 0 & 0 \\ 0 & I_{m-t} \end{bmatrix}$, we then write $G_t = \begin{bmatrix} L_{m-t} & U_t \\ U_t & L_{m-t} \end{bmatrix}$. Similar to part 1 above, the circuit for g_t is simply H applied to each of the first t qubits. Although this is a special case where the Hadamard operator was applied to consecutive qubits, we note that the symplectic transformation for Hadamards applied to arbitrary nonconsecutive qubits can be derived in a similar fashion.

Hence, we have demonstrated the elementary symplectic transformations in $\mathrm{Sp}(2m, \mathbb{F}_2)$ that are associated with arbitrary Hadamard, Phase, Controlled-Z and Controlled-NOT gates. Since we know that these gates, along with HW_N, generate the full Clifford group [16], these elementary symplectic transformations form a universal set corresponding to physical operators in the Clifford group.

APPENDIX B: PROOF OF THEOREM 1

Let $F = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$ so that $[A \ B] \Omega [A \ B]^T = 0$ and $[C \ D] \Omega [C \ D]^T = 0$ since $F \Omega F^T = \Omega$. We will perform a sequence of row and column operations to transform F into the form $\Omega T_{R_1} \Omega$ for some symmetric R_1. If $\mathrm{rank}(A) = k$ then there exists a row transformation Q_{11}^{-1} and a column transformation Q_2^{-1} such that $Q_{11}^{-1} A Q_2^{-1} = \begin{bmatrix} I_k & 0 \\ 0 & 0 \end{bmatrix}$. Using the notation for elementary symplectic transformations discussed above, we apply Q_{11}^{-1} and $A_{Q_2^{-1}}$ to $[A \ B]$ and obtain

$$\begin{bmatrix} Q_{11}^{-1} A & Q_{11}^{-1} B \end{bmatrix} \begin{bmatrix} Q_2^{-1} & 0 \\ 0 & Q_2^T \end{bmatrix} = \begin{bmatrix} I_k & 0 & R_k & E' \\ 0 & 0 & E & B_{m-k} \end{bmatrix}$$

$$\triangleq [A' \quad B'],$$

where B_{m-k} is an $(m - k) \times (m - k)$ matrix. Since the above result is again the top half of a symplectic matrix, we have

$[A'\ B']\, \Omega\, [A'\ B']^T = 0$ which implies R_k is symmetric, $E = 0$ and hence rank$(B_{m-k}) = m - k$. Therefore we determine an invertible matrix Q_{m-k} which transforms B_{m-k} to I_{m-k} under row operations. Then, we apply $Q_{12}^{-1} \triangleq \begin{bmatrix} I_k & 0 \\ 0 & Q_{m-k} \end{bmatrix}$

$$\begin{bmatrix} Q_{12}^{-1}Q_{11}^{-1}A & Q_{12}^{-1}Q_{11}^{-1}B \end{bmatrix} \begin{bmatrix} Q_2^{-1} & 0 \\ 0 & Q_2^T \end{bmatrix}$$

$$= \begin{bmatrix} I_k & 0 & R_k & E' \\ 0 & 0 & 0 & I_{m-k} \end{bmatrix}.$$

Now, we observe that we can apply row operations to this matrix and transform E' to 0. We left multiply by $Q_{13}^{-1} \triangleq \begin{bmatrix} I_k & E' \\ 0 & I_{m-k} \end{bmatrix}$:

$$\begin{bmatrix} Q_{13}^{-1}Q_{12}^{-1}Q_{11}^{-1}A & Q_{13}^{-1}Q_{12}^{-1}Q_{11}^{-1}B \end{bmatrix} \begin{bmatrix} Q_2^{-1} & 0 \\ 0 & Q_2^T \end{bmatrix}$$

$$= \begin{bmatrix} I_k & 0 & R_k & 0 \\ 0 & 0 & 0 & I_{m-k} \end{bmatrix}.$$

Since the matrix $R_2 \triangleq \begin{bmatrix} R_k & 0 \\ 0 & 0 \end{bmatrix}$ is symmetric, we apply the elementary transformation T_{R_2} from the right to obtain

$$\begin{bmatrix} I_k & 0 & R_k & 0 \\ 0 & 0 & 0 & I_{m-k} \end{bmatrix} \begin{bmatrix} I_k & 0 & R_k & 0 \\ 0 & I_{m-k} & 0 & 0 \\ 0 & 0 & I_k & 0 \\ 0 & 0 & 0 & I_{m-k} \end{bmatrix} = \begin{bmatrix} I_k & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{m-k} \end{bmatrix}.$$

Finally, we apply the elementary transformation $G_k \Omega = \begin{bmatrix} U_k & L_{m-k} \\ L_{m-k} & U_k \end{bmatrix}$ to obtain

$$\begin{bmatrix} I_k & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{m-k} \end{bmatrix} \begin{bmatrix} I_k & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{m-k} \\ 0 & 0 & I_k & 0 \\ 0 & I_{m-k} & 0 & 0 \end{bmatrix} = \begin{bmatrix} I_k & 0 & 0 & 0 \\ 0 & I_{m-k} & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} I_m & 0 \end{bmatrix}.$$

Hence we have transformed F to the form $\Omega\, T_{R_1} \Omega = \begin{bmatrix} I_m & 0 \\ R_1 & I_m \end{bmatrix}$, i.e., if we define $Q_1^{-1} \triangleq Q_{13}^{-1}Q_{12}^{-1}Q_{11}^{-1}$ then we have

$$A_{Q_1^{-1}} F A_{Q_2^{-1}} T_{R_2} G_k \Omega = \Omega\, T_{R_1} \Omega.$$

Rearranging terms and noting that $A_Q^{-1} = A_{Q^{-1}}$, $\Omega^{-1} = \Omega$, $G_k^{-1} = G_k$, $T_{R_2}^{-1} = T_{R_2}$ we obtain $F = A_{Q_1} \Omega\, T_{R_1} G_k T_{R_2} A_{Q_2}$. ∎

APPENDIX C: MATLAB CODE FOR ALGORITHM 2

```matlab
function F_all = find_all_symp_mat(U,V,I,J)

I = I(:)';
J = J(:)';
Ibar = setdiff(1:m,I);
Jbar = setdiff(1:m,J);
alpha = length(Ibar) + length(Jbar);
tot = 2^(alpha*(alpha+1)/2);
F_all = cell(tot,1);

% Find one solution using symplectic
    transvections (Algorithm 1)
F0 = find_symp_mat(U([I, m+J], :), V);

A = mod(U * F0, 2);
Ainv = gf2matinv(A);
IbJb = union(Ibar,Jbar);
Basis = A([IbJb, m+IbJb],:); % these rows
    span the subspace W^{\perp} in Theorem 7
Subspace =
    mod(de2bi((0:2^(2*length(IbJb))-1)',
            2*length(IbJb)) * Basis, 2);

% Collect indices of free vectors in the
    top and bottom halves of Basis
% Note: these are now row indices of Basis,
    not row indices of A!!
[~, Basis_fixed_I, ~] = intersect(IbJb,I);
    % intersect(IbJb,I) = intersect(I,Jbar)
[~, Basis_fixed_J, ~] = intersect(IbJb,J);
    % intersect(IbJb,J) = intersect(Ibar,J)
Basis_fixed = [Basis_fixed_I, length(IbJb)
    + Basis_fixed_J];
Basis_free = setdiff(1:2*length(IbJb),
    Basis_fixed);

Choices = cell(alpha,1);

% Calculate all choices for each free
    vector using just conditions imposed
% by the fixed vectors in Basis (or
    equivalently in A)
for i = 1:alpha
    ind = Basis_free(i);
    h = zeros(1,length(Basis_fixed));
    % Impose symplectic inner product of 1
        with the "fixed" symplectic pair
    if (i <= length(Ibar))
        h(Basis_fixed == length(IbJb) + ind)
            = 1;
    else
        h(Basis_fixed == ind - length(IbJb))
            = 1;
    end
    % Check the necessary conditions on the
        symplectic inner products
    Innpdts = mod(Subspace *
        fftshift(Basis(Basis_fixed,:), 2)',
        2);
    Choices{i,1} = Subspace(bi2de(Innpdts)
        == bi2de(h), :);
end
```

```
% First free vector has 2^(alpha) choices,
    second has 2^(alpha-1) choices and so on
for l = 0:(tot - 1)
   Bl = A;
   W = zeros(alpha,2*m); % Rows are choices
       made for free vectors
   % W(i,:) corresponds to
       Basis(Basis_free(i),:)
   lbin =
       de2bi(l,alpha*(alpha+1)/2,'left-msb');
   v1_ind =
       bi2de(lbin(1,1:alpha),'left-msb') + 1;
   W(1,:) = Choices{1,1}(v1_ind,:);
   for i = 2:alpha
       % vi_ind loops through the
           2^(alpha-(i-1)) valid choices for
           the i-th free vector
      vi_ind =
          bi2de(lbin(1,sum(alpha:-1:alpha-(i-2))
              + (1:(alpha-(i-1)))),'left-msb') +
              1;
      Innprods = mod(Choices{i,1} *
          fftshift(W,2)', 2);
      % Impose symplectic inner product of
          0 with chosen free vectors
      h = zeros(1,alpha);
      % Handle case when Basis contains a
          symplectic pair of free vectors
      if (i > length(Ibar))
          h(Basis_free == Basis_free(i) -
              length(IbJb)) = 1;
      end
      % Check the necessary and sufficient
          conditions on the symplectic inner
          products
      Ch_i = Choices{i,1}(bi2de(Innprods)
          == bi2de(h), :);
      W(i,:) = Ch_i(vi_ind,:); % use the
          vi_ind-th valid choice for the
          i-th free vector
   end
   Bl([Ibar, m+Jbar], :) = W; % replace
       rows of free vectors with current
       choices
   F = mod(Ainv * Bl, 2);  % this is the
       matrix F' in Theorem 7
   F_all{l+1,1} = mod(F0 * F, 2);
end

end
```

REFERENCES

[1] N. Rengaswamy, R. Calderbank, S. Kadhe, and H. D. Pfister, "Synthesis of logical Clifford operators via symplectic geometry," in *Proc. IEEE Int. Symp. Inf. Theory*. 2018, pp. 791–795, doi: 10.1109/ISIT.2018.8437652.

[2] P. W. Shor, "Scheme for reducing decoherence in quantum computer memory," *Phys. Rev. A*, vol. 52, no. 4, pp. R2493–R2496, 1995, doi: 10.1088/0256-307X/14/7/003.

[3] A. R. Calderbank and P. W. Shor, "Good quantum error-correcting codes exist," *Phys. Rev. A*, vol. 54, pp. 1098–1105, Aug. 1996, doi: 10.1103/PhysRevA.54.1098.

[4] A. M. Steane, "Simple quantum error-correcting codes," *Phys. Rev. A*, vol. 54, no. 6, pp. 4741–4751, 1996, doi: 10.1103/PhysRevA.54.4741.

[5] R. Calderbank, E. Rains, P. Shor, and N. Sloane, "Quantum error correction via codes over GF(4)," *IEEE Trans. Inf. Theory*, vol. 44, no. 4, pp. 1369–1387, Jul. 1998, doi: 10.1109/18.681315.

[6] D. Gottesman, "Stabilizer codes and quantum error correction," Ph.D. dissertation, Dept. Phys., California Inst. Technol., Pasadena, CA, USA, 1997. [Online]. Available: https://arxiv.org/abs/quant-ph/9705052

[7] D. Bacon, "Operator quantum error-correcting subsystems for self-correcting quantum memories," *Phys. Rev. A*, vol. 73, 2006, Art. no. 012340, doi: 10.1103/PhysRevA.73.012340.

[8] T. J. Yoder, "Universal fault-tolerant quantum computation with Bacon-Shor codes," 2017. [Online]. Available: https://arxiv.org/pdf/1705.01686

[9] D. Gottesman and I. L. Chuang, "Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations," *Nature*, vol. 402, no. 6760, pp. 390–393, 1999, doi: 10.1038/46503.

[10] S. Bravyi and J. Haah, "Magic-state distillation with low overhead," *Phys. Rev. A*, vol. 86, no. 5, 2012, Art. no. 052329, doi: 10.1103/PhysRevA.86.052329.

[11] S. Bravyi and A. Kitaev, "Universal quantum computation with ideal Clifford gates and noisy ancillas," *Phys. Rev. A*, vol. 71, no. 2, 2005, Art. no. 022316, doi: 10.1103/PhysRevA.71.022316.

[12] M. M. Wilde, "Logical operators of quantum codes," *Phys. Rev. A*, vol. 79, no. 6, 2009, Art. no. 062322, doi: 10.1103/PhysRevA.79.062322.

[13] T. Brun, I. Devetak, and M.-H. Hsieh, "Correcting quantum errors with entanglement," *Science*, vol. 314, no. 5798, pp. 436–439, 2006, doi: 10.1126/science.1131563.

[14] T. A. Brun, I. Devetak, and M.-H. Hsieh, "Catalytic quantum error correction," *IEEE Trans. Inf. Theory*, vol. 60, no. 6, pp. 3073–3089, Jun. 2014, doi: 10.1109/TIT.2014.2313559.

[15] A. R. Calderbank, E. M. Rains, P. W. Shor, and N. J. A. Sloane, "Quantum error correction and orthogonal geometry," *Phys. Rev. Lett.*, vol. 78, no. 3, pp. 405–408, 1997, doi: 10.1103/PhysRevLett.78.405.

[16] D. Gottesman, "An introduction to quantum error correction and fault-tolerant quantum computation," 2009. [Online]. Available: http://arxiv.org/abs/0904.2557

[17] D. Litinski, "A game of surface codes: Large-scale quantum computing with lattice surgery," *Quantum*, vol. 3, p. 128, Mar. 2019, doi: 10.22331/q-2019-03-05-128.

[18] V. Kliuchnikov, D. Maslov, and M. Mosca, "Asymptotically optimal approximation of single qubit unitaries by Clifford and T circuits using a constant number of ancillary qubits," *Phys. Rev. Lett.*, vol. 110, no. 19, May 2013, Art. no. 190502, doi: 10.1103/PhysRevLett.110.190502.

[19] M. Amy, D. Maslov, M. Mosca, and M. Roetteler, "A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits," *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 32, no. 6, pp. 818–830, Jun. 2013, doi: 10.1109/TCAD.2013.2244643.

[20] D. Maslov and M. Roetteler, "Shorter stabilizer circuits via Bruhat decomposition and quantum circuit transformations," *IEEE Trans. Inf. Theory*, vol. 64, no. 7, pp. 4729–4738, Jul. 2018, doi: 10.1109/TIT.2018.2825602.

[21] A. Fagan and R. Duncan, "Optimising Clifford circuits with quantomatic," *Electron. Proc. Theor. Comput. Sci.*, vol. 287, pp. 85–105, 2019, doi: 10.4204/EPTCS.287.5.

[22] R. Iten, O. Reardon-Smith, L. Mondada, E. Redmond, R. S. Kohli, and R. Colbeck, "Introduction to UniversalQCompiler," 2019. [Online]. Available: http://arxiv.org/abs/1904.01072

[23] R. Duncan, A. Kissinger, S. Pedrix, and J. van de Wetering, "Graph-theoretic simplification of quantum circuits with the ZX-calculus," *Quantum*, vol. 4, p. 279, 2020, doi: 10.22331/q-2020-06-04-279.

[24] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, "Surface codes: Towards practical large-scale quantum computation," *Phys. Rev. A*, vol. 86, no. 3, 2012, Art. no. 032324, doi: 10.1103/PhysRevA.86.032324.

[25] M. Grassl and M. Roetteler, "Leveraging automorphisms of quantum codes for fault-tolerant quantum computation," in *Proc. IEEE Int. Symp. Inf. Theory*. Jul. 2013, pp. 534–538, doi: 10.1109/ISIT.2013.6620283.

[26] A. Kubica and M. E. Beverland, "Universal transversal gates with color codes: A simplified approach," *Phys. Rev. A*, vol. 91, no. 3, 2015, Art. no. 032330, doi: 10.1103/PhysRevA.91.032330.

[27] R. Chao and B. W. Reichardt, "Fault-tolerant quantum computation with few qubits," *npj Quantum Inf.*, vol. 4, no. 1, 2018, Art. no. 42, doi: 10.1038/s41534-018-0085-z.

[28] T. Can, N. Rengaswamy, R. Calderbank, and H. D. Pfister, "Kerdock codes determine unitary 2-designs," *IEEE Trans. Inf. Theory*, vol. 66, no. 10, pp. 6104–6120, Oct. 2020, doi: 10.1109/TIT.2020.3015683.

[29] J. Combes, C. Granade, C. Ferrie, and S. T. Flammia, "Logical randomized benchmarking," 2017. [Online]. Available: http://arxiv.org/abs/1702.03688

[30] E. Magesan, J. M. Gambetta, and J. Emerson, "Characterizing quantum gates via randomized benchmarking," *Phys. Rev. A*, vol. 85, no. 4 2012, Art. no. 042311, doi: 10.1103/PhysRevA.85.042311.

[31] P. Murali, J. M. Baker, A. J. Abhari, F. T. Chong, and M. Martonosi, "Noise-adaptive compiler mappings for noisy intermediate-scale quantum computers," in *Proc. 24th Int. Conf. Archit. Support Program. Lang. Oper. Syst.*, 2019, pp. 1015–1029, doi: 10.1145/3297858.3304075.

[32] J. Preskill, "Quantum Computing in the NISQ era and beyond," 2018. [Online]. Available: https://arxiv.org/abs/1801.00862

[33] A. Paler, "On the influence of initial qubit placement during NISQ circuit compilation," in *Quantum Technol. and Optimization Problems*, S. Feld and C. Linnhoff-Popien, Eds. Berlin, Germany: Springer, 2019, pp. 207–217, doi: 10.1007/978-3-030-14082-3_18.

[34] Y. Shi, N. Leung, P. Gokhale, Z. Rossi, D. I. Schuster, H. Hoffmann, and F. T. Chong, "Optimized compilation of aggregated instructions for realistic quantum computers," in *Proc. 24th Int. Conf. Archit. Support Program. Lang. Oper. Syst.*, 2019, pp. 1031–1044, doi: 10.1145/3297858.3304018.

[35] P. Murali, A. Javadi-Abhari, F. T. Chong, and M. Martonosi, "Formal constraint-based compilation for noisy intermediate-scale quantum systems," *Microprocess. Microsyst.*, vol. 66, pp. 102–112, 2019, doi: 10.1016/j.micpro.2019.02.005.

[36] B. Nash, V. Gheorghiu, and M. Mosca, "Quantum circuit optimizations for NISQ architectures," *Quantum Sci. Technol.*, vol. 5, no. 2, 2020, Art. no. 025010, doi: 10.1088/2058-9565/ab79b1.

[37] R. Harper and S. T. Flammia, "Fault-tolerant logical gates in the IBM quantum experience," *Phys. Rev. Lett.*, vol. 122, no. 8, 2019, Art. no. 080504, doi: 10.1103/PhysRevLett.122.080504.

[38] N. M. Linke *et al.*, "Fault-tolerant quantum error detection," *Sci. Adv.*, vol. 3, no. 10, Oct. 2017, Art. no. e1701074, doi: 10.1126/sciadv.1701074.

[39] R. Ozeri, "The trapped-ion qubit tool box," *Contemp. Phys.*, vol. 52, no. 6, pp. 531–550, 2011, doi: 10.1080/00107514.2011.603578.

[40] Y. Nam *et al.*, "Ground-state energy estimation of the water molecule on a trapped-ion quantum computer," *npj Quantum Inf.*, vol. 6, no. 1, p. 33, 2020, doi: 10.1038/s41534-020-0259-3.

[41] N. M. Linke *et al.*, "Experimental comparison of two quantum computing architectures," *Proc. Nat. Acad. Sci.*, vol. 114, no. 13, pp. 3305–3310, 2017, doi: 10.1073/pnas.1618020114.

[42] J. Haah, M. B. Hastings, D. Poulin, and D. Wecker, "Magic state distillation with low space overhead and optimal asymptotic input count," *Quantum*, vol. 1, p. 31, 2017, doi: 10.22331/q-2017-10-03-31.

[43] J. Haah and M. B. Hastings, "Codes and protocols for distilling *T*, controlled-*S*, and Toffoli gates," *Quantum*, vol. 2, p. 71, 2017, doi: 10.22331/q-2018-06-07-71.

[44] C. Gidney and A. G. Fowler, "Efficient magic state factories with a catalyzed |CCZ> to 2|T> transformation," *Quantum*, vol. 3, p. 135, 2018, doi: 10.22331/q-2019-04-30-135.

[45] B. Zeng, A. Cross, and I. L. Chuang, "Transversality versus universality for additive quantum codes," *IEEE Trans. Inf. Theory*, vol. 57, no. 9, pp. 6272–6284, Sep. 2011, doi: 10.1109/TIT.2011.2161917.

[46] B. Eastin and E. Knill, "Restrictions on transversal encoded quantum gate sets," *Phys. Rev. Lett.*, vol. 102, no. 11, 2009, Art. no. 110502, doi: 10.1103/PhysRevLett.102.110502.

[47] M. Newman and Y. Shi, "Limitations on transversal computation through quantum homomorphic encryption," *Quantum Inf. Comput.*, vol. 18, nos. 11/12, pp. 0927–0948, Apr. 2018, doi: 10.5555/3370220.3370223.

[48] S. Bravyi, D. Gosset, and R. König, "Quantum advantage with shallow circuits." *Sci.*, vol. 362, no. 6412, pp. 308–311, 2018, doi: 10.1126/science.aar3106.

[49] S. Bravyi, D. Gosset, R. Koenig, and M. Tomamichel, "Quantum advantage with noisy shallow circuits," *Nat. Phys.*, 2020, to be published, doi: 10.1038/s41567-020-0948-z.

[50] D. Gottesman, "Theory of fault-tolerant quantum computation," *Phys. Rev. A*, vol. 57, no. 1, pp. 127–137, Jan 1998, doi: 10.1103/PhysRevA.57.127.

[51] N. Rengaswamy, R. Calderbank, S. Kadhe, and H. D. Pfister, "Synthesis of Logical Clifford Operators via Symplectic Geometry," 2018. [Online]. Available: http://arxiv.org/abs/1803.06987v1

[52] N. Rengaswamy, R. Calderbank, and H. D. Pfister, "Unifying the Clifford hierarchy via symmetric matrices over rings," *Phys. Rev. A*, vol. 100, no. 2, 2019, Art. no. 022304, doi: 10.1103/PhysRevA.100.022304.

[53] A. Ekert and C. Macchiavello, "Quantum error correction for communication," *Phys. Rev. Lett.*, vol. 77, no. 12, pp. 2585–2588, Sep. 1996, doi: 10.1103/PhysRevLett.77.2585.

[54] C. Chamberland, P. Iyer, and D. Poulin, "Fault-tolerant quantum computing in the Pauli or Clifford frame with slow error diagnostics," *Quantum*, vol. 2, p. 43, 2018, doi: 10.22331/q-2018-01-04-43.

[55] T. Can, "The Heisenberg-Weyl group, finite symplectic geometry, and their applications," Senior thesis, Dept. Mathematics, Duke Univ., Durham, NC, USA, May 2018.

[56] J. Dehaene and B. De Moor, "Clifford group, stabilizer states, and linear and quadratic operations over GF(2)," *Phys. Rev. A*, vol. 68, no. 4, Oct. 2003, Art. no. 042318, doi: 10.1103/PhysRevA.68.042318.

[57] Z. Webb, "The Clifford group forms a unitary 3-design," *Quantum Inf. Comput.*, vol. 16, nos. 15/16, pp. 1379–1400, 2016. doi: 10.5555/3179439.3179447.

[58] P. Diaconis and M. Shahshahani, "The subgroup algorithm for generating uniform random variables," *Prob. Eng. Inf. Sci.*, vol. 1, no. 1, pp. 15–32, 1987, doi: 10.1017/S0269964800000255.

[59] R. Koenig and J. A. Smolin, "How to efficiently select an arbitrary Clifford group element," *J. Math. Phys.*, vol. 55, no. 12, Dec. 2014, Art. no. 122202, doi: 10.1063/1.4903507.

[60] P. W. Jones, A. Osipov, and V. Rokhlin, "A randomized approximate nearest neighbors algorithm," *Appl. Comput. Harmon. Anal.*, vol. 34, no. 3, pp. 415–444, 2013, doi: 10.1016/j.acha.2012.07.003.

[61] K. Choromanski, F. Fagan, C. Gouy-Pailler, A. Morvan, T. Sarlos, and J. Atif, "TripleSpin - a generic compact paradigm for fast machine learning computations," 2016. [Online]. Available: http://arxiv.org/abs/1605.09046.

[62] N. Rengaswamy, R. Calderbank, M. Newman, and H. D. Pfister, "On optimality of CSS codes for transversal *T*," *IEEE J. Sel. Areas Inf. Theory*, to be published. [Online]. Available: http://arxiv.org/abs/1910.09333

[63] A. Salam, E. Al-Aidarous, and A. E. Farouk, "Optimal symplectic householder transformations for SR decomposition," *Lin. Algebra Its Appl.*, vol. 429, pp. 1334–1353, 2008, doi: 10.1016/j.laa.2008.02.029.

[64] T. Can, "An Algorithm to Generate a Unitary Transformation from Logarithmically Many Random Bits," Research Independent Study, Dept. Mathematics, Duke Univ., Durham, NC, USA, 2017.

[65] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge, U.K.: Cambridge Univ. Press, 2010.