Trellis Decoding for Qudit Stabilizer Codes and Its Application to Qubit Topological Codes

ERIC SABO1, ARUN B. ALOSHIOUS2 (Member, IEEE), AND KENNETH R. BROWN2,3,4,5 (Senior Member, IEEE)
1School of Mathematics, Georgia Institute of Technology, Atlanta, GA 30332 USA
2Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708 USA
3School of Chemistry and Biochemistry, Georgia Institute of Technology, Atlanta, GA 30332 USA
4Department of Physics, Duke University, Durham, NC 27708 USA
5Department of Chemistry, Duke University, Durham, NC 27708 USA

Corresponding author: Arun B. Aloshious (e-mail: aloshious@iitg.ac.in).

The work of Eric Sabo was supported by the National Science Foundation through a Quantum Information Science and Engineering Network fellowship under Award DMR-17474266. This work was supported in part by the Intelligence Advanced Research Projects Activity, Office of the Director of National Intelligence, through an Army Research Office Contract W911NF-16-1-0082, in part by the Army Research Office under Grant W911NF-21-1-0005, and in part by the Army Research Office Multidisciplinary University Research Initiative under Grant W911NF-18-1-0218 and Grant W911NF-16-1-0349.

ABSTRACT Trellis decoders are a general decoding technique first applied to qubit-based quantum error correction codes by Ollivier and Tillich in 2006. Here, we improve the scalability and practicality of their theory, show that it has strong structure, extend the results using classical coding theory as a guide, and demonstrate a canonical form from which the structural properties of the decoding graph may be computed. The resulting formalism is valid for any prime-dimensional quantum system. The modified decoder works for any stabilizer code \( S \) and separates into two parts: 1) a one-time offline computation that builds a compact graphical representation of the normalizer of the code, \( S^\perp \) and 2) a quick, parallel, online query of the resulting vertices using the Viterbi algorithm. We show the utility of trellis decoding by applying it to four high-density length-20 stabilizer codes for depolarizing noise and the well-studied Steane, rotated surface, and 4.8.8/6.6.6 color codes for \( Z \) only noise. Numerical simulations demonstrate a 20% improvement in the code-capacity threshold for color codes with boundaries by avoiding the mapping from color codes to surface codes. We identify trellis edge number as a key metric of difficulty of decoding, allowing us to quantify the advantage of single-axis (\( X \) or \( Z \)) decoding for Calderbank–Steane–Shor codes and block decoding for concatenated codes.

INDEX TERMS Quantum error correction, Trellis decoding.

I. INTRODUCTION

Quantum error correction allows for the in-principle preservation of quantum information in the presence of noise by encoding the information into a larger quantum system. The utility of any quantum error correction approach relies not only on the code parameters but also on the existence of good decoders. There are presently few generic decoders applicable to random quantum error correcting codes. This is not without reason; quantum decoding is known to be computationally more difficult than classical decoding [1], [2]. Most decoders developed so far are specifically tailored to a given code family based on intuitive, visual, or physical arguments. Many decoders and decoding schemes in the literature are unique to the simulation they are presented with, while others, especially those for topological codes, have enjoyed widespread use, study, and success.

For topological codes, such as toric, rotated surface, and color codes, the full decoding problem is typically reduced to that of finding a minimum-weight error, usually under an uncorrelated noise model, and is modeled as a perfect matching for the nonzero syndrome. The Blossom V implementation of Edmond’s minimum-weight perfect matching (MWPM) algorithm is most often used [3], [4], but other approaches, such as greedy matching, have also been applied [5]. These are applicable to topological codes whenever the syndrome and errors have a string-like pattern. Despite
the deceptive similarities to the surface codes, applying the same techniques to the color codes results instead in a hypergraph matching problem [6], which has no efficient algorithm. To get around this, some color code decoders first map the problem to two or more copies of a toric code that are then decoded and the information pieced back together to form a correction for the original code [7], [8], [9], [10], [11]. Recently, an approach to apply MWPM in codes with certain properties has been studied in [12]. The union find decoder [13], [14], [15], [16] has been successfully applied to surface codes and homological product codes, but a color code or more general stabilizer code implementation has still yet to be developed. Other popular decoders for topological codes include those based on cellular automata [17], [18], [19], integer programming [20], renormalization [21], [22], [23], and utilizing standard puzzles [24]. Topological codes are often simulated on infinite lattices or finite lattices with periodic boundary conditions, and decoders sometimes require structural features, such as locality and translational invariance.

Of particular interest are decoders that apply to multiple families of codes with little to no modification up to input data. Belief propagation [25], [26], tensor network [27], [28], [29], [30], [31], [32], and machine learning (ML) [33], [34], [35], [36], [37], [38] based decoders generally fall into this category. Belief propagation is useful when codes satisfy specific sparsity properties but is inherently more difficult for quantum than classical codes. Recent work has improved this by introducing a common classical postprocessing step to prevent the decoder from getting stuck in loops [39], [40]. Tensor network decoders are theoretically exact maximum likelihood decoders but remain practically limited by computation with finite resources. As with much of ML, ML decoders trade good performance with a thorough understanding of its decisions and the theoretical guarantees that come with it.

One benefit to general decoding techniques is that they remove things, such as geometric or topological constraints, that complicate algorithms. Ollivier and Tillich [41] introduced a decoding algorithm in 2006 called trellis decoding. This is applicable to any qubit stabilizer code $S$, but only a $[[5, 1, 1]]$ code was explored. This is a port of a highly successful and well-understood classical decoder and works by building a highly compact and efficient graphical representation of the algebraic structure of the normalizer of $S$, $\mathcal{S}^\perp$, called a trellis. The trellis contains all the valid combinations of logical operators and stabilizer generators that will return the system to its code space. Decoding proceeds by using dynamical programming to globally search for the minimal weight path in the trellis corresponding to the measured syndrome. This is performed efficiently in exactly $n$ major steps for an $[[n, k, d]]$ stabilizer code, although the amount of work required in each step varies with respect to a predictable code-dependent formula and can be significant depending on the amount of available resources. Beyond the brief introduction in [41], this decoder was discussed in [42] and [43] for quantum convolutional codes, and a portion of the algorithm was improved in [44]. Many fundamental questions and theoretical properties remained unanswered.

As previously described in the literature, trellis decoding was simply not practical as it required the repeated processing of potentially massive amounts of data (all of the elements of $S^\perp$). A simple observation made in this article allows us to reduce the number of times these data are processed to just once. However, this is still unpractical as enumerating the elements of $S^\perp$ may not be possible for many interesting codes. Here, we expand on the previous literature by fully developing the theoretical foundations of the decoder. The outcome is a way to extract all the information needed to construct the trellis solely from the generators of $S$ and $S^\perp$. This object is independent of the error model and may be computed once and saved for future simulations. The main goal and overall contribution of this article is in making trellis decoding practical. To date, it has been the most general and widely applicable decoding technique in quantum error correction.

Nonbinary (qudit) stabilizer codes are less explored and understood than qubit codes, yet potentially offer significant computational advantages. Of particular interest here is that qudit codes offer improved error thresholds over their qubit counterparts [45], [46], [47], [48]. Nonbinary decoders are lacking as many standard qubit decoders are not applicable to even direct generalizations of qubit codes without significant modification, if at all possible [45], [46], [47], [48], [49], [50]. Although qudit codes are not simulated in this work, we notably extend the theory for any prime dimension in Section III, and the decoder works for both qubit and qudit stabilizer codes without any need for higher dimensional modifications.

We begin by summarizing the main ingredients and fundamental results of our work in Section II. A rigorous theory is developed in Section III, while proofs of the numerous technical lemmas are relegated to Appendix D to increase readability. We attempt to mimic the classical coding theory literature as closely as possible to make this work available to the widest audience. It is perhaps remarkable that many of our results have classical analog despite the differences between classical and quantum codes forcing alternative proof techniques. We view this as a strength of the overall theory of trellis decoding and point out similarities and differences, providing references to the classical literature, whenever possible. This work could be seen as an extension of the classical theory to codes over $\mathbb{F}_p \times \mathbb{F}_p$ given appropriate adjustments to the proofs.

In Section III-C, we adopt a classical quantitative metric for the difficulty of decoding a given stabilizer code based on the structure of the trellis. This allows us to show that the color codes are fundamentally more difficult to decode than the rotated surface codes, and the 4.8.8 color code is more difficult to decode than the 6.6.6 color code. We also make quantitative arguments as to exactly how much easier it is to decode a Calderbank–Shor–Steane (CSS) code using
independent $X$ and $Z$ decoders versus a single decoder for the full code. Although not unique to trellis decoding, we show that large gains are made when splitting the trellis into $X$ and $Z$ parts.

Numerical results for code-capacity (memory model) simulations are presented in Section V. First, we demonstrate the power of the theory by decoding four high-density codes from the quantum error-correcting code table at [51]. As we are unaware of any other decoding algorithm for these codes, it is difficult to quantify the performance of the trellis with these data. We, therefore, second, proceed to apply our work to the rotated surface and color code families for which there already exist numerous (and some highly optimized) decoders. Notably, the color codes (with boundary) are natively decoded without reference to other codes or notions of color, homology, matching, boundaries, lifting, projections, restrictions, charges, excitations, strings, and/or mappings. Trellis decoding for stabilizer codes is optimal minimum weight (in contrast to classical trellis decoding which is maximum likelihood), and numerical results for $Z$ only noise should match that of MWPM for the surface codes and exceed the current best thresholds in the literature for the color codes, although we only simulate codes up to distances 17–21. Finally, we simulate a two-stage suboptimal decoder for the level-2 concatenated Steane code.

II. SYNDROME TRELLIS

Let $p$ be a prime. A quantum bit, called a qubit if $p = 2$ and a qudit if $p > 2$, is a normalized vector in $\mathbb{C}^p$. An $n$ qudit system is a Hilbert space $\mathbb{C}^{DQ}_n \equiv \mathbb{C}^p \otimes \cdots \otimes \mathbb{C}^p$. Let $[\{i\} | i \in \mathbb{F}_p]$ and consider the standard $X$ and $Z$ operators defined by $X^a[i] = [i + a]$ and $Z^b[i] = \omega^{b[i]}$, where $\omega$ is a primitive $2p$th root of unity if $p$ is even or odd, respectively, and “+” represents modulo $p$ addition. Denote the generalized Pauli group on $n$ qudits by $\mathcal{P}_n = \{a^*X^aZ^b : a, b \in \mathbb{F}_p^n, c \in \mathbb{F}_p\}$. We will refer to elements in the form $IXXIXZY$ as Pauli strings to distinguish them from any of their numerical representations. A stabilizer group is an abelian subgroup $\mathcal{S} \subseteq \mathcal{P}_n$. The quantum stabilizer code $\mathcal{Q}$ is the shared eigenspace of the operators of $\mathcal{S}$ (or their representations), which we may assume without loss of generality to be the +1 eigenspace: $\mathcal{Q} \equiv \{v \in \mathbb{C}_n^DQ | Sv = v \forall S \in \mathcal{S}\}$. We assume that dim $\mathcal{Q} > 1$ and that the number of nonidentity Pauli operators of any element in $\mathcal{S}$ is at least two. The centralizer of $\mathcal{S}$ in $\mathcal{P}_n$, $\mathcal{C}_{\mathcal{P}_n}(\mathcal{S})$, is all the elements of $\mathcal{P}_n$ that commute with all elements of $\mathcal{S}$. The stabilizer code associated with $\mathcal{Q}$ is the classical code defined by mapping $\mathcal{S}/Z(\mathcal{P}_n)$ to $\mathbb{F}_2^{DQ}$, where $Z(\mathcal{P}_n) = \{a^*X^a | c \in \mathbb{F}_p\}$ is the center of $\mathcal{P}_n$. We will always assume $\omega^k = 1$ in this work and so will not distinguish between the stabilizer group and code, viewing the latter as a mere representation of the former. Therefore, we will denote both the objects by $\mathcal{S}$. The dual of $\mathcal{S}$ is defined to be $\mathcal{S}^\perp \equiv \mathcal{C}_{\mathcal{P}_n}(\mathcal{S})/Z(\mathcal{P}_n)$ and contains all stabilizers and logical Pauli operators for $\mathcal{Q}$. This is the dual code of the stabilizer code $\mathcal{S}$ with respect to the symplectic inner product and has dimension $n + k$ viewed as a subspace of $\mathbb{F}_2^{DQ}$, where $k = \dim \mathcal{S}$.

The classical coding theory literature contains several definitions for the trellis of a linear block code. Some of them, such as the Bahl–Cocke–Jelinek–Raviv (BCJR) [52], the Wolf [53], and the Forney–Muder [54, 55] trellises, are now known to be isomorphic. With the benefit of this hindsight, Ollivier and Tillich [41] demonstrated how to port what is often known as the syndrome trellis to the quantum setting. They referred to their construction as the Wolf trellis; however, the discussion here more closely follows that of BCJR. As such, we simply use syndrome trellis in this work, although, since unlike classical coding theory, there is only one definition of a trellis for quantum error correction, we may just as well shorten this to simply trellis. One may attempt to port the other classical trellis to the quantum case, but the definition used in [41] and this work is known to be minimal in a rigorous sense that will be defined in Section III, rendering alternate definitions undesirable. The interested reader is referred to the tutorial piece [56] for a general discussion of classical trellises.

Recall that a directed edge, $e$, in a graph goes from source, $s(e)$, to terminus, $t(e)$. We represent the operators of $\mathcal{S}$ as $s(e)$, $t(e)$, where $P$ is a label of the form $X^aZ^b$ for $a, b \in \mathbb{F}_p$.

Definition 1 ((Quantum) SYNDROME TRELLIS): A trellis for an $[n, k, d]_p$ stabilizer code $\mathcal{Q}$ is a directed multigraph with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$ such that the following statements hold.

1. There are $n + 1$ disjoint sets of vertices $\mathcal{V}_i$ with $V = \mathcal{V}_0 \sqcup \cdots \sqcup \mathcal{V}_n$ and $|\mathcal{V}_0| = |\mathcal{V}_n| = 1$.
2. There are $n$ disjoint sets, $E_i$, of directed edges from $\mathcal{V}_{i-1}$ to $\mathcal{V}_i$ with $E = E_1 \sqcup \cdots \sqcup E_n$.
3. Each vertex $v \in \mathcal{V}_i$ has a unique label given by an $(n - k)$-tuple of syndromes, although the same label may be present in multiple $\mathcal{V}_i$.
4. Each edge $e \in E_i$ is a unique triple of the form $(s(e), P, t(e))$, where $P$ is a label of the form $X^aZ^b$ for $a, b \in \mathbb{F}_p$.
5. Each edge is assigned a weight $w(t(e)) \in \mathbb{R}^+ \cup \{\infty\}$.

Following the classical literature, vertices are referred to as states, $\mathcal{V}_i$ as state spaces, and $\mathcal{V}_i$ is said to be at depth $i$. The edge sets $E_i$ are referred to as the $i$th section and the edges as branches. The condition that edge labels must be unique between a fixed pair of source and terminus vertices means that our trellises are proper. Trellises that do not satisfy this condition are called improper and are not considered in this work. It is perhaps more convenient to define the trellis without weights ($v$), but we stick with decades of precedent in including them here.

To construct the syndrome trellis, choose a set of stabilizer generators $\{S_1, \ldots, S_{n-1}\}$ for the stabilizer group $\mathcal{S}$ and fix this set throughout. For $P \in \mathcal{P}_n$, let a subscript $P$ denote the $i$th element of this product.

Definition 2: Let $\mathcal{J} \subset \{0, \ldots, n\}$ be an index set. Then, define $\pi_{\mathcal{J}} : \mathcal{P}_n \rightarrow \mathcal{P}_n$ to be the map, which projects all Pauli operators at indices $\mathcal{J}^c = \{0, \ldots, n\} \setminus \mathcal{J}$ to the identity, where $\pi_0(P)$ is the all identity Pauli string.
We introduce 0 in the index set to represent empty set. Thus, \( \pi_0 \) can be represented as \( \pi_0 \). This notation will help understanding the definition of past and future sets defined in Section III using simple notations.

To avoid ambiguities with which qudit receives the relative phase, assume that \( \omega' = 1 \). Let \( \sigma_i : P_n \to \mathbb{F}_p \) be the map from a Pauli string to its syndrome

\[
\sigma_i(P) = (\langle S_1, \pi_{[0,\ldots,i]}(P) \rangle, \ldots, \langle S_{n-k}, \pi_{[0,\ldots,i]}(P) \rangle) \tag{1}
\]

where \( \langle , \cdot \rangle \) is an appropriate inner product, and let \( P_s \) be a Pauli string with syndrome \( s \) with respect to the chosen generators, \( s = \sigma_n(P_s) \). Compute \( S^{\perp} \) and then \( P_s S^{\perp} \). Since everything in \( S^{\perp} \) has zero syndrome, everything in \( P_s S^{\perp} \) has syndrome \( s \).

The vertices in each set are labeled by the values \( \sigma_i(P) \) for all \( P \in P_s S^{\perp} \)

\[
V_i = \{ \sigma_i(P) : 0 \leq i \leq n, P \in P_s S^{\perp} \}. \tag{2}
\]

A directed edge is created from the vertex \( \sigma_{i-1}(P) \in V_{i-1} \) to the vertex \( \sigma_i(P) \in V_i \) and labeled with the \( i \)th component of \( P \)

\[
E_i = \{ (\sigma_{i-1}(P), P_i, \sigma_i(P)) : P \in P_s S^{\perp} \}. \tag{3}
\]

We show in Section III that the vertices and edges are highly constrained. At depth and section \( i \), there are (see Theorem 1) \( |V_i| = p^\dim S^{\perp} - \dim S^{\perp}_{i-1} \dim S^{\perp}_{i} \) vertices and \( |E_i| = p^\dim S^{\perp} - \dim S^{\perp}_{i-1} - \dim S^{\perp}_{i} \) edges. The groups \( S^{\perp}_{F_i} \) and \( S^{\perp}_{D_i} \) depend on \( S^{\perp} \) and are defined in later sections (see

\[1\] Up until now, we have considered \( S^{\perp} \) as a set of Pauli strings and not as a subset of \( P^n \). One may define an equivalent vector space over the strings by defining scalar multiplication as the raising of a Pauli operator to a power.
Section III). Every vertex in \( V_i \) has the same incoming degree, \( \text{deg}_{\text{in}}(v) = p^\text{dim}S_{v_i}^\perp - \text{dim}S_{v_{i-1}} \), and outgoing degree, \( \text{deg}_{\text{out}}(v) = p^\text{dim}S_{v_i}^\perp - \text{dim}S_{v_{i+1}} \), degrees (see Corollary 2), and only certain patterns of edge connections are allowed (see Theorem 2). For a fixed qudit ordering, the structure of the trellis is invariant of the choice of stabilizer or logical generators (see Lemma 3), but a specific choice of generators, the trellis-oriented form (TOF), allows us to immediately read-off the dimensions of the sets in the previous formulas (see Proposition 4).

We take the trellis for \( S^\perp \) (with respect to the zero syndrome) to be the fundamental object in this work. The general decoding scheme proceeds as follows: 1) construct the trellis; 2) measure a syndrome and shift the base trellis with respect to it; 3) decode the shifted trellis; and 4) repeat steps 2) and 3). We avoid assigning edge weights to the trellis until step 2) so the trellis resulting from step 1) is independent of any measured syndromes or error model. Hence, step 1) is an offline procedure, while steps 2) and 3) are online. Trellises for larger distance codes may, therefore, be computed once and saved for future use. For the codes considered in this work, the efficiency of the construction algorithm made this unnecessary for most distances. Since \( |V| \leq |E| \), shifting the trellis is \( \Theta(|E|) \) in time, but the procedure is easily parallelized. As we will see later, decoding is also \( \Theta(|E|) \) and may also be parallelized.

With the edge weights prescribed above, the desired correction is given by the maximum-likely (or minimum-weight) path from \( V_0 \) to \( V_n \). We refer to this in the following as the “optimal path.” The Viterbi algorithm is an example of forward dynamical programming and is the most common trellis-based decoder [57]. The idea is as follows. Choose an arbitrary vertex \( v \in V_i \) for some \( i \neq 0 \) or \( n \), and suppose that the optimal path travels through \( v \). Then, the optimal path can be split into two parts: the optimal path going from \( V_0 \) to \( v \) and the optimal path going from \( v \) to \( V_n \). Compute the optimal path from \( V_0 \) to \( v \) and repeat for all \( v \). Once the optimal paths are computed for every vertex in \( V_{i-1} \), the optimal path for a vertex in \( V_i \) is chosen by finding the minimum value of the sums of the incoming edge weights plus the weight of the optimal path for each edge’s source vertex in \( V_{i-1} \). The weight at \( V_0 \) is arbitrary but is convenient to initialize to zero. Having completed this for all vertices, the Pauli correction may be read off the trellis by taking the edge labels for the optimal path connecting \( V_0 \) to \( V_n \). See Appendix A Fig. 8; for a visual example of the Viterbi algorithm run on Fig. 1; an example implementation is also provided. The algorithm is often described in a manner in which vertices with no outgoing edges are removed from the system. This creates nice diagrams, but actual deletion steps can be algorithmically expensive and should be ignored as in Fig. 1. Also note that the partial syndromes, or vertex labels, make no appearance in the algorithm and may be safely removed after the construction phase, also ignoring vertices in the shifting phase. The partial syndromes of the distance 15 rotated surface code are 224 bits long, for example, and not storing them, even in a more efficient fashion, leads to large savings in storage.

The Viterbi algorithm is sometimes confused with other standard minimum-path graph algorithms, but it is distinct and more efficient given the strict edge and vertex dependencies required in the definition of a trellis. Underlying the success of this algorithm is the implicit assumption that the trellis behaves as a Markov chain, with each \( V_i \) only depending on the result at \( V_{i-1} \). In particular, at each \( V_i \), all the information regarding the correction of all previous qudits has already been processed.

The behavior of the decoder depends strongly on the assumed error model. If all Pauli operators are equally likely, as in the standard depolarizing noise model, the Viterbi algorithm acts as a minimum Hamming weight decoder and may run into a significant number of ties, which may be broken arbitrarily. For a biased noise model, the decoder is able to differentiate between, say, \( X \) and \( Z \) and is able to make a more informed choice.

III. PROPERTIES

Throughout this section, we denote the vertex with zero syndrome by \( \emptyset \). Dimension will always refer to the number of generators of an object, and \( |\cdot| \) will be reserved for cardinality. A Pauli string will be denoted by \( P \), whereas \( p \) will denote the characteristic of the underlying field. We first show several properties of the syndrome (base) trellis; then, we turn to TOF before finally discussing the Viterbi algorithm. Many, but not all, of the results here have classical analogies, and we attempt to provide original citations to the best of our knowledge for those that are not typically discussed in textbooks. Classically, however, codes are vector spaces, whereas here we consider stabilizer codes as groups. This changes the techniques, but we try to maintain the same overall direction of the proofs if possible. We consider it a strength of the theory that a single framework can handle both classical and quantum trellises. The closest classical work to this is that of Forney and Trott [58] on convolutional group codes.

A. SYNDROME TRELLIS

The first result was understood and implicitly used in [41] and [44] but was never explicitly stated. It is perhaps obvious, but we include a proof here for completeness.

**Proposition 1:** There is a one-to-one correspondence between Pauli strings in \( S^\perp \) and length-\( n \) paths in the trellis.

**Corollary 1:** Let \( v_i \in V_i \) and \( v_{i+1} \in V_{i+1} \) be arbitrary. If there is a Pauli element, \( P_i \), which takes \( v_i \) to \( v_{i+1} \), then \( (v_i, P_i, v_{i+1}) \in E_i \).

In the previous section, we attempted to stick to the notation of the preceding literature, following [41]. At this point, we make some adjustments that we find necessary to simplify the proofs and discussion. The main reason for this is that \( S \) and \( S^\perp \) are groups, whereas \( V_i \) and \( E_i \) are vector spaces. Promoting the Pauli strings to vectors or the vector spaces...
to groups provides for a more coherent argument. Choosing Pauli strings over their symplectic representation, we henceforth introduce the following notation. We begin with a rather general definition for completeness; however, our goal and use case throughout this article is the rather natural splitting of the qudit indices into a “past” and “future,” which is defined later in Definition 4.

Definition 3:

1) Let $\mathcal{J} \subset \{0, \ldots, n\}$ be an index set and define $A_{\mathcal{J}}$ to be the set of Pauli strings in a subgroup $A \leq \mathcal{P}_n$ whose Pauli operators are equal to the identity on the complement of $\mathcal{J}$, $\mathcal{J}^c := \{0, \ldots, n\} \setminus \mathcal{J}$. Both the definitions are conceptually equivalent.

Remark: If $C$ is a classical code, then $C_{\mathcal{J}}$ and $C_{|\mathcal{J}|}$ are the shortened and punctured codes of $C$ with respect to $\mathcal{J}$, respectively. The concepts of shortening and puncturing are a bit more complicated for quantum codes [59], so we refrain from using this terminology here.

Example 1: Let $\mathcal{J} = \{0, \ldots, i\}$. Then, $A_{\mathcal{J}}$ is the set Pauli strings, which naturally have identity elements in indices $\{i+1, \ldots, n\}$ and $A_{\mathcal{J}^c} = \{1\}$ since $\pi_{\mathcal{J}^c}(A) = \{1\}$ in $\mathcal{P}_n$, and $A_{\mathcal{J}^c}$ is the set of Pauli strings whose elements in indices $\{i+1, \ldots, n\}$ have been projected to the identity regardless of their initial value.

Critical to our proofs is the fact that since $\pi_{\mathcal{J}}(PP') = \pi_{\mathcal{J}}(P)\pi_{\mathcal{J}}(P')$ for $P, P' \in \mathcal{P}_n$, $\pi_{\mathcal{J}}$ is a group homomorphism. Then, ker $\pi_{\mathcal{J}}(A) = A_{\mathcal{J}^c}$ and ker $\pi_{\mathcal{J}^c}(A) = A_{|\mathcal{J}|}$, and $A_{\mathcal{J}^c} \subseteq A$ and $A_{|\mathcal{J}|} \subseteq A$ as all kernels are normal. (Recall that the notation $A \triangleleft B$ means that $A$ is a normal subgroup of $B$.) The product group $A_{\mathcal{J}}A_{|\mathcal{J}|}$ is also normal, and as $\mathcal{J} \cap \mathcal{J}^c = \emptyset$, $A_{\mathcal{J}}A_{|\mathcal{J}|} = A_{\mathcal{J}} \times A_{|\mathcal{J}|} \subseteq A$.

In keeping with the standard trellis literature, we introduce the following further terminology.

Definition 4: Fix an index $i \in \{0, \ldots, n\}$. Then, the past and future, with respect to $i$, are defined to be the index sets $p_i := \{0, \ldots, i\}$ and $f_i := \{i+1, \ldots, n\}$, respectively.

In particular, we will utilize the sets of Pauli strings $S_{p_i}, S_{i}, S_{p_i}^\perp, S_{i}^\perp$, and likewise for $S_{\perp}$. See Fig. 2 for a summary of the relationships between the groups. Note that the literature varies on which set includes the index $i$. In this work, we index the qudits on $\{1, \ldots, n\}$, but it is necessary to include zero in order to project to the identity.

The previously mentioned subgroups (see Fig. 2) depend only on the set $A$ and not the choice of generators, although $S$ and $S_{\perp}$ may have different subgroups at the same index. Lagrange’s theorem restricts their cardinality to powers of $p$. (In fact, this was a motivation for promoting $V$ and $E$ to groups instead of dealing with less-constrained vector subspace dimensions.) Initially, $A_{p_i} = A$ and monotonically decreases in size as $i$ goes to $n$. $A_{i} = (I^\otimes n)$. Likewise, $A_{p_0} = (I^\otimes n)$ and monotonically increases in size as $i$ goes to $n$. $A_{p_0} = A$

For a fixed $i$, sort the elements of $S_{\perp}$ into sets $S_{p_i}^\perp, S_{i}^\perp$, and $S_{|i|}^\perp$, where the “active” set, denoted by $a$, contains the remaining elements neither wholly in the past nor future. Include the identity in $S_{|i|}^\perp$ to give this trivial intersection.

An element of $S_{\perp}$ may, hence, be decomposed in the form $P\omega_{p_0}P_{v}$, where $P \in S_{p_i}^\perp, P_o \in S_{i}^\perp$, and $P_v \in S_{|i|}^\perp$. An element $P_{v,i} \in S_{|i|}^\perp$ is of the form $P_{v,i} = (P_{p_0})_{p_0}(P_{|i|})_{|i|}$, and similarly for the future. Thus, $S_{p_0}^\perp = \text{span}(S_{p_0}^\perp, (S_{a_i}^\perp)_{p_0})$ and $S_{i}^\perp = \text{span}(S_{i}^\perp, (S_{a_i}^\perp)_{i})$. Fix a Pauli string $P \in S_{a_i}^\perp$, and let $v$ be the vertex generated by $P$ in $V_i$. Then, every length-$i$ path from $V_0$ to $v$ is generated by the coset $P_{|i|}S_{|i|}^\perp$. Likewise, every path from $v$ to $V_n$ is generated by an element of the coset $S_{p_0}^{\perp}P_{p_0}^{\perp}$. Putting these together, every path from $V_0$ to $V_n$ may be viewed as a coset $P_{p_0}^{\perp}S_{p_0}^{\perp}P_{|i|}S_{|i|}^{\perp}$ with respect to $v$. See Fig. 3 for a visual summary.

As mentioned in the previous section, there are historically numerous definitions of trellises in the classical literature. It was, therefore, important to determine whether some are “better” than others, leading to the concept of a minimal trellis (defined below). We will see below that if another
trelis would be defined for stabilizer codes, the trelis of this work is minimal. The following proposition takes us in this direction and holds for any definition of a trelis for a stabilizer code.

**Proposition 2 (See [41, Lemma 1]):** Given a stabilizer code $S$, any trelis for $S$ must satisfy

$$|V_i| \geq p^{\dim S_i \dim S_{\pi_i} \dim S_{\eta_i}}$$

$$|E_i| \geq p^{\dim S_i \dim S_{\eta_{i-1}} \dim S_{\eta_i}}.$$

**Definition 5:** A trelis that meets the lower bounds of Proposition 2 is minimal.

**Remark:** Nonminimal trelises will not be discussed in this work. As a trivial example, one could define a trelis where each element of $S^2$ consists of its own path from $V_0$ to $V_n$ without intersecting any other path. The number of vertices at each depth would then be $|S^2|$. We refer the reader to [56] for further (classical) examples. The concept of a minimal trelis comes from [55].

While the edges of the trelis encode $S^2$, the vertices do not take the logical operators into account and are determined by $S$, an important point in sharp contrast with the classical case whose vertices and edges are both constructed relative to the same object: the code.

For the vertices, fix $i$, let $P$ be an arbitrary element of $S^2$, and let $S_j$ be a fixed generator of $S$. If $S_j \in S_{\pi_i}$, then $\langle S_j, \pi_{(i+1,\ldots,n)}(P) \rangle = 0$ since $S_j$ is the identity at indices $\{i+1,\ldots,n\}$ and these will commute with any elements of $P$ at these positions. Thus, the syndrome of $P$ with respect to the generator $S_j$ has already been completely determined (and is zero since $P \in S^2$). Likewise, if $S_j \in S_{\eta_i}$, then $\langle S_j, \pi_{(0,\ldots,i)}(P) \rangle = 0$ since $S_j$ is the identity at indices $\{0,\ldots,i\}$, which, again commute with $\pi_{(0,\ldots,i)}(P)$. Only generators $S_j \in S_{\eta_i}$ can have nonzero syndromes.

**Theorem 1 (Quantum SPACE THEOREMS):** The syndrome trelises of Section 2 is minimal, i.e.,

1) (Quantum state-space theorem; see [41, Lemma 2])

$$|V_i| = p^{\dim S_i \dim S_{\pi_i} \dim S_{\eta_i}}. \quad (4)$$

2) (Quantum branch space theorem)

$$|E_i| = p^{\dim S_i \dim S_{\eta_{i-1}} \dim S_{\eta_i}}. \quad (5)$$

**Remark:** The sequences $\{|V_i|\}_{i=0}^n$ and $\{|E_i|\}_{i=1}^n$ are often referred to as the state space and branch space complexity profiles, respectively. The quantity $\max_i \dim V_i$ is the state complexity, $\max_i \dim E_i$ is the branch complexity, and $|E|$ is the edge complexity. We will not utilize these in this work.

**Lemma 1:** The set of edges in $E_i$ with terminal vertex $\overline{S}$ is a subgroup of $E_i$.

**Corollary 2:** For $1 \leq i \leq n$, every vertex $v \in V_i$ has incoming degree

$$\deg_{\text{in}}(v) = p^{\dim S_{\eta_i} \dim S_{\eta_{i-1}}}. \quad (6)$$

and for $0 \leq i \leq n-1$, every vertex $v \in V_i$ has outgoing degree

$$\deg_{\text{out}}(v) = p^{\dim S_{\eta_i} \dim S_{\eta_{i+1}}}. \quad (7)$$

**Lemma 2:** Let $A = S$ or $S^2$. Then, $0 \leq \dim A_i - \dim A_{i+1} \leq 2$ for $0 \leq i \leq n-1$ and $0 \leq \dim A_{\pi_i} - \dim A_{\eta_{i+1}} \leq 2$ for $1 \leq i \leq n$.

**Corollary 3:** For $0 \leq i \leq n-1$, $\deg_{\text{out}}(v) \in \{1, p, p^2\}$, and for $1 \leq i \leq n$, $\deg_{\text{in}}(v) \in \{1, p, p^2\}$. Equivalently, for $1 \leq i \leq n$, $\dim V_i \leq \dim V_0 \leq 2i$.

Table 1 records this result graphically for the special case $p = 2$. It is well known in classical trelis theory that the dimension change in Lemma 2 is bounded by one [60]. This is due to classical codes only having one symbol alphabet instead of the two $(X)$ and $(Z)$, which allows one to completely row reduce to a single pivot per column. Any configuration in Table 1 with a two is, therefore, unique to the quantum setting. The next theorem shows that these are the only possible configurations. No corresponding proof exists in the classical literature.

**Theorem 2:** Consider an arbitrary edge $e = (s(e), P, t(e)) \in E_i$ and define $\mathcal{I}_{i-1} = \{v \in V_{i-1} : \exists e' \in E_i, t(e') = t(e)\}$ and $\mathcal{I}_i = \{v \in V_i : \exists e' \in E_i, s(e') = s(e)\}$. Then, the vertices of $\mathcal{I}_{i-1}$ and $\mathcal{I}_i$ form a completely connected bipartite graph, and no other elements of $V_i \setminus \mathcal{I}_{i-1}$ or $V_i \setminus \mathcal{I}_i$ are connected to the vertices in $\mathcal{I}_i$ and $\mathcal{I}_{i-1}$, respectively. If there exists a parallel edge in $\mathcal{I}_i$, then all edges in $E_i$ are parallel with the same number of edges in parallel.

**Corollary 4:** The number of disjoint edge configurations in $E_i$ is $p^{\dim S_{\pi_i} - \dim S_{\eta_{i+1}} - \dim S_{\eta_i}}$.

This is simply $|V_{i+1}|/\deg_{\text{out}}(v)$ for $v \in V_i$.

At first glance, (4) should be viewed with skepticism. It is common for $\dim S_{\pi_i} = \dim S_{\eta_i} = 0$ (see Example 4), and if the same holds true for $S^2$, then $|V_i| = p^{n+k}$, which is notably greater than the number of total possible syndromes, $p^{n-k}$. Applying the classical formula for $V_i$ or using the logic above, one would expect $\dim V_i = \dim S - \dim S_{\pi_i} - \dim S_{\eta_i}$, and in fact, this also works. The inclusion of the logicals, therefore, always forces $\dim S_{\pi_i} + \dim S_{\eta_i} \geq 2k$. The various formulas throughout this section may, therefore, be written with this alternative view of $\dim V_i$, but this leads to a lack of cancellation of terms and an unpleasant factor of
TABLE 1. Various Possible Edge Configurations for p = 2

| Δ dim S_{p,i} | Δ dim S_{q,i} | E_i | Δ dim S_{p,i} | Δ dim S_{q,i} | E_i | Δ dim S_{p,i} | Δ dim S_{q,i} | E_i |
|----------------|----------------|-----|----------------|----------------|-----|----------------|----------------|-----|
| 0              | 0              | 1   | 0              | 1              | 0   | 2              | 0              | 0   |
| 0              | 1              | 1   | 1              | 1              | 1   | 2              | 1              | 0   |
| 0              | 2              | 1   | 2              | 2              | 2   | 2              | 2              | 2   |

Since Δdim S_{p,i} and Δdim S_{q,i} are bounded by one in classical theory, any configurations with a two is unique to the quantum setting. Here, Δ represents the change in dimension from i – 1 to i or i to i + 1, as appropriate.

2k floating around whose lack of obvious effect on the results requires justification.

Corollary 5: For 0 ≤ i ≤ n, dim S_{p,i} + dim S_{q,i} ≥ 2k.

The next result says that for a fixed qudit ordering, the choice of stabilizer or logical generators is irrelevant.

Lemma 3: Any two minimal trellises for the same stabilizer code are isomorphic.

B. TRELLIS-ORIENTED FORM

The term “trellis-oriented” was introduced in passing in an appendix by Forney [54] and was not thoroughly defined until seven years later by Kschischang and Sorokine [61]. Here, we adapt the latter approach to the quantum setting, as did [41], but we will stick closer to the classical notation than that of [41]. The term “minimal-span” is preferred by some authors in the literature.

Let \( P \in P_n \) be an arbitrary Pauli string with components \( P_i \) at index \( i \).

Definition 6:

1) The left index of \( P, L(P) \), is the smallest index such that \( P_i \neq I \) and the right index of \( P, R(P) \), is the largest index such that \( P_i \neq I \).
2) The span of \( P \) is the index set \( \{ L(P), L(P) + 1, \ldots, R(P) \} \) and the span length of \( P \) is the cardinality of the span of \( P, R(P) - L(P) + 1 \). The span of the identity string is defined to be \( \{ \} \) and the corresponding span length to be 0.
3) Say \( P \) is active at depth \( i \) if \( i - 1 \) and \( i \) are in the span of \( P \), i.e., \( L(P) \leq i - 1 \) and \( R(P) \geq i \).

Definition 7 (Left–RIGHT PROPERTY): A set of Pauli strings is said to have the left–right property if no two elements with the same left or right index have more than one power of \( X \) or \( Z \) at these index locations.

The next definition is intended to be applied to a set of generators and not the entirety of its span.

Definition 8 (TOF): A set of Pauli strings is said to be in TOF if the sum of the span lengths of the elements is as small as possible.

Proposition 3: A set of Pauli strings is in TOF if and only if it has the left–right property.

Example 2: The generators for the rotated surface and color codes naturally have the left–right property. The canonical form of the stabilizer generators of the \([5, 1, 3]\) code as cyclic shifts of \( XZZXI \) may be put into TOF as follows:

\[
\begin{align*}
S_1 &= XZZXI \\
S_2 &= IZZZX \\
S_3 &= XIZZX \\
S_4 &= ZIXZX \\
S_2S_3 &= IZYZZ \\
S_1S_3 &= XZZXI \\
S_1S_4 &= ZZXI \\
S_1S_2 &= ZXXZ \\
S_2S_3 &= IZYYZ \\
S_1S_3 &= IZZXZ \\
S_1S_4 &= ZYZI \\
S_2S_3 &= IZZZX \\
S_1S_4 &= ZZXI \\
\end{align*}
\]

The left and right indices for each row are, in order, \([1, 4], [1, 4], [2, 5], [2, 5]\), where we have used the notation \([L(P), R(P)]\). It is customary, and sometimes included in the definition, to rearrange the strings by increasing left index when viewing them in a matrix-like format, similar to a row-reduced matrix, but this is not strictly necessary.

The procedure used in the previous example is quite simple: after reducing the left side from top to bottom, the right side is reduced from bottom to top and the upper stabilizer is
always replaced instead of the lower. This prevents the multiplication of the stabilizers from changing the left indices. The key to automating this is considering Pauli strings as length-n vectors over \( \mathbb{F}_p \), and using appropriate operations in this field. This should be compared with the complex algorithm given in [44] for obtaining the TOF using the symplectic representation over \( \mathbb{F}_p \).

The quantum TOF is more complicated than in classical theory. To gain some intuition for this, consider a set of Pauli strings in TOF. A quantum index, either left or right, is of the form \( X^a Z^b \), and \( X^{a'} Z^{b'} \) with all exponents nonzero, one can only eliminate the other to produce an identity if and only if they are scalar multiples of each other in \( \mathbb{F}_p \), i.e., \( (a', b') \in (a, b) \in \mathbb{Z}_p \times \mathbb{Z}_p \). In the more general case, by repeated application of the string to itself, a and b can individually be made to take any value in \( \mathbb{F}_p \), i.e., Pauli strings are linear over \( \mathbb{F}_p \). In particular, \( -a' \in (a) = \mathbb{Z}_p \) since \( p \) is prime, and likewise for \( b \). So, one of \( a' \) or \( b' \) may always be eliminated. An index \( X^a \) cannot eliminate an index \( Z^b \) and vice versa. To summarize, the possibilities for left or right indices for qubit codes are \( \{X, I\}, \{Z, I\}, \{Y, I\}, \{X, Z\}, \{Y, X\}, \{Y, Z\} \). This is in stark contrast to the classical case where a matrix can always be put into reduced-row echelon form.

Let \( P \) and \( P' \) be two Pauli strings with the left–right property and span \([L, R]\); then, \( PP' \) also has span \([L, R]\). This is often referred to as the predicatable span property. If \( P \) and \( P' \) have span \([L, R_1]\) and \([L, R_2]\), respectively, with \( R_1 \prec R_2 \), then \( PP' \) has span \([L, R_2]\), and likewise for \([L, R_1]\) and \([L, R_2]\). These two cases are not possible classically when there is only one symbol alphabet, necessitating different proof strategies. 

**Example 3:** The first two stabilizers for the TOF of the following [[8, 2, 3]] code [51] have the same left index but different right indices:

\[
\begin{align*}
Y & \ Z & Z & X & I & I & I \\
Z & I & I & X & Y & I & I \\
I & X & Z & Z & Y & X & Y \\
I & Z & Y & I & Y & X & I \\
I & I & X & Y & X & Z & Y \\
I & I & Z & I & I & Y & X
\end{align*}
\]

If we start with a generating set for \( A \), the resulting TOF still generates \( A \). Each generator has order \( p \) and generates \( p-1 \) nonidentity strings with the same span. Suppose that two generators \( P \) and \( P' \) have spans \([L_1, R_1]\) and \([L_2, R_2]\), respectively, and both \( R_1 \leq R_2 \leq i \in \{1, \ldots, n\} \). Then, all products of the \( p \) elements generated by \( P \) and the elements generated by \( P' \) also have right index less than or equal to \( i \).

If there are no other generators with right index less than or equal to \( i \), then these are all of the elements of \( A_p \). Extending this argument shows the following.

**Proposition 4:** Let \( A \) be a set of Pauli strings with generators \( A_i \) in TOF. Then

\[
\dim A_{p_i} = | \{ A_i | R(A_i) \leq i \} | \quad (8)
\]

\[
\dim A_i = | \{ A_i | L(A_i) \geq i + 1 \} | \quad (9)
\]

**Example 4:** The stabilizers of the [[5, 1, 3]] code appear in Example 2. The stabilizers of \( S^\perp \) have a different TOF

\[
\begin{align*}
X & \ Y & X & I \\
I & X & Y & X & I \\
I & I & X & Y & X & I \\
I & I & Z & X & Z & I \\
I & I & Z & X & Z \\
\end{align*}
\]

Applying Proposition 4 to both sets, we get Table 2.

| \( [5, 1, 3] \) Code | \( V_i \) | \( E_i \) |
|-------------------|---------|------|
| 0 0 0 0 0 0 4 4 4 4 | 1 1 1 1 1 1 1 1 1 1 | 0 0 0 0 0 0 0 0 0 0 |
| 1 0 2 0 4 4 4 4 4 4 | 1 1 1 1 1 1 1 1 1 1 | 0 0 0 0 0 0 0 0 0 0 |
| 0 0 0 0 0 0 1 1 1 1 | 0 0 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 0 0 |
| 0 0 0 0 0 0 2 2 2 2 | 0 0 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 0 0 |
| 0 0 0 0 0 0 3 3 3 3 | 0 0 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 0 0 |
| 0 0 0 0 0 0 4 4 4 4 | 0 0 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 0 0 |
| 0 0 0 0 0 0 5 5 5 5 | 0 0 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 0 0 |

Unfortunately, the previous argument also shows that the TOF is not unique since a generator may be replaced by any of its \( p-1 \) multiples. (Example 2, replace \( S_1 S_3 \) with, for example, \( S_1 S_2 S_3 \)). However, in light in of Theorem 1 and Corollary 2, the entire structure of the trellis may be read-off directly from the generators of \( S^\perp \) when put into TOF. Proposition 4 also provides trivial proofs of results such as Lemma 2 by merely counting the number of possible new generators obtained when shifting indices in TOF.

**C. VITERBI ALGORITHM**

It is worth clarifying exactly what the trellis solves. Let \( E \in \mathcal{P}_n \) be a Pauli error acting on a quantum stabilizer code with the stabilizer group \( S = \langle S_1, \ldots, S_{n-k} \rangle \). The syndrome, \( s = (s_1, \ldots, s_{n-k}) \), is the ordered tuple of commutation relations of \( E \) with the stabilizer elements, \( s_j = \langle S_j, E \rangle \), where the inner product is chosen appropriately. Since \( S \leq S_p(s) \leq \mathcal{P}_n \), \( E \) is of the form \( E = S I T_s \), where \( S \in S \) is a stabilizer, \( L \in S_p(s) \), \( S \) is a logical operator, and \( T_s \in S_P \cap S_p(s) \) is a pure error corresponding to the measured syndrome \( s \). One can always write down a pure error, which matches the given syndrome, so we may assume that \( T_s \) is known.

The standard decoding problem in the stabilizer formalism is to find the most likely error

\[
\hat{E}^{ML}(s) = \arg \max_{E \in \mathcal{P}_n} \Pr (L, S | T_s)
\]

\[
= \arg \max_{E \in \mathcal{P}_n} \frac{\Pr (L, S, T_s)}{\Pr (T_s)}
\]
where the last equality holds because \( \Pr(T_i) \) is just a positive constant. This is suboptimal because it searches for the most likely error instead of the most likely equivalence class of errors defined by the sum of the probabilities of each element in the coset. A decoder taking the degeneracy of errors into account would, therefore, solve

\[
\hat{E}^{DM}(s) = \arg\max_{L \in \mathcal{C}_{T_n}(S)/S} \Pr(L | T_n)
\]

\[
= \arg\max_{L \in \mathcal{C}_{T_n}(S)/S} \sum_{S \in S} \Pr(L, S | T_n). \tag{11}
\]

Note that for the depolarizing channel \( \mathcal{E}(\rho) = (1 - p)\rho + \frac{p}{3}X \rho X + \frac{p}{3}Y \rho Y + \frac{p}{3}Z \rho Z \), we have

\[
\hat{E}^{ML}(s) = \arg\max_{L \in \mathcal{P}_n} \Pr(L, S, T_i)
\]

\[
= \arg\max_{E \in \mathcal{P}_n} \Pr(|E|) (1 - p)^{n - |E|} \tag{12}
\]

where \( \text{wt} \) denotes the Hamming weight of the Pauli operator.

After measuring a syndrome, constructing the pure error, and shifting the trellis, the paths are in one-to-one correspondence with all elements of the form \( SL_T \). Thus, finding the minimum-weight path through the trellis solves (10), and the trellis is an exact maximum-likelihood decoder. We show in an upcoming paper how to use the trellis to solve (11). Pelchat and Poulin [43] present a different procedure for this in their investigation of trellis decoding for quantum convolutional codes.

**Proposition 5:** The Viterbi algorithm solves (10) for independently and identically distributed noise models.

**Remark:** For independent \( X, Z \), or depolarizing channels, the weights of the edges of the trellis are in correspondence with the Hamming weight of the vector, in which case the Viterbi algorithm returns the minimum Hamming weight correction. This is the same solution returned by algorithms such as MWPM.

As mentioned in Section II, the full decoding procedure consists of finding a pure error, shifting the trellis, and then applying the Viterbi algorithm. The shifting procedure requires at most \( n \) symplectic inner products followed by \( |V| + |E| \) shifts. Fortunately, this operation is embarrassingly parallel, and the runtime may be significantly reduced depending on the implementation and available resources. For a potentially large computational speedup, one may skip shifting the vertices, as they play no role in the Viterbi algorithm. It is clear from Algorithm 1 that every edge incurs a single addition followed by incoming-degree comparisons for every vertex. For large codes, the vertices of \( V_i \) may be processed independently in parallel. The Viterbi algorithm may also be run simultaneously in a “forward pass” from \( V_0 \) to \( V_i \) and a “backward pass” from \( V_n \) to \( V_{i+1} \). The choice of the optimal splitting depends highly on the left–right balance of the trellis. Further optimizations, such as those based on the coset structure of the code (vertical symmetry), are generally code specific. In practice, the shifting and decoding can proceed one after the other inside a given section before moving onto the next, but for the sake of theoretical fundamentals, we momentarily assume that they are separate.

**Theorem 3 (See [56, Th. 2.10]):** The Viterbi algorithm requires \( \Theta(|E|) \) arithmetic operations.

Theorem 1 shows that the syndrome trellis minimizes \(|V|\) and \(|E|\), but in light of the previous proof, it would be useful to show that it also minimizes the quantity \(|E| - |V| + 1\). That is, the trellis minimizes the number of algorithmic operations. Classically, Muder [55] first showed that the analogous trellis minimizes \(|V|\), McEliece [56], [62] showed it minimizes \(|E|\), and Vardy and Kschischang [63] showed that it minimizes \(|E| - |V| + 1\). The proof of this follows for the quantum case with minor modifications (see Proposition 6), and we include it here in Theorem 4 for completeness. We first establish a geometric interpretation of this quantity.

The quantity \(|E|\) may be further resolved using the edge configurations in Table 1. While these are drawn for the special case \( p = 2 \), here, we will use them to represent the corresponding diagrams for higher \( p \). One may check that substituting \( p = 2 \) produces the correct coefficients in the following formula. We have

\[
|E| = 1 \cdot \#(\begin{array}{c}
\end{array}) + p \cdot \#(\begin{array}{c}
\end{array}) + p^2 \cdot \#(\begin{array}{c}
\end{array}) + p^3 \cdot \#(\begin{array}{c}
\end{array}) + p^4 \cdot \#(\begin{array}{c}
\end{array})
\]

\[
(13)
\]
Similarly, ignoring $V_0$ and counting the right-hand sides, we obtain
\[ |V| - 1 = 1 \cdot \#(\text{square}) + 1 \cdot \#(\text{triangle}) + 1 \cdot \#(\text{pentagon}) + p \cdot \#(\text{hexagon}) + \cdots \]
\[ + p^2 \cdot \#(\text{heptagon}) + \cdots . \]

**Definition 9:** A vertex $v$ with $\deg_{\text{out}}(v) > 1$ is called an expansion and a merger if $\deg_{\text{in}}(v) > 1$.

The number of mergers is graphically given by
\[ M = (p - 1) \cdot \#(\text{square}) + (p^2 - 1) \cdot \#(\text{triangle}) + \cdots \]
\[ + p(p - 1) \cdot \#(\text{hexagon}) + p^2(p - 1) \cdot \#(\text{heptagon}) + \cdots . \]
\[ + (p^2 - 1) \cdot \#(\text{octagon}) + p(p^2 - 1) \cdot \#(\text{nonagon}) + \cdots . \]
\[ + p^2(p^2 - 1) \cdot \#(\text{decagon}) + \cdots . \]

(14)

Subtracting (14) from (13) shows that this is exactly $|E| - |V| + 1$ [64], as expected from the proof of Proposition 3. To match the classical proof in [63], we switch from mergers to expansions $E$. Repeating the arguments of the proof with $\deg_{\text{in}}(v)$ in place of $\deg_{\text{out}}(v)$ shows $E = |E| - |V| + 1$ as well. This is intuitively clear from the fact that the trellis both starts and ends with a single vertex. Writing a similar expression to (15) for expansions and setting $E = M$, the diagrams that share a left–right symmetry cancel leaving only an equality of total asymmetric edge configurations.

Returning to our goal, it is clear from Section III that the number of paths from $V_0$ to any vertex in $V_1$ is $p^{\dim S_{1i}^\perp}$ and the total number of paths from $V_0$ to all of $V_i$ is $p^{\dim S_{1i}^\perp}$. Furthermore, for any definition of a trellis, the number of paths is at most $p^{\dim S_{1i}^\perp}$ and the total number of paths is at least $p^{\dim S_{1i}^\perp}$.

**Lemma 4 (See [63, Lemma 5]):** Let $\mathcal{P}_i(v)$ denote the number of paths from $V_0$ to $v \in V_i$ and $\mathcal{P}_i = \sum_{v \in V_i} \mathcal{P}_i(v)$ be the number of paths from $V_0$ to all vertices in $V_i$. Then
\[ \mathcal{P}_i = 1 + \sum_{j=0}^{i-1} \sum_{v \in V_j} \mathcal{P}_j(v) (\deg_{\text{out}}(v) - 1). \]

The next result is used but not proved in [63]. We present the proof for the quantum case; the proof of their original expression follows from this by restricting to $\dim S_{p_i}^\perp - \dim S_{p_{i+1}}^\perp \leq 1$. Recall that $(\dim S_{p_i}^\perp)$ is an increasing sequence whose value changes when a new right index is encountered in the TOF. Let $\{R_i\}$ be the locations of the right indices. The idea behind the following proof is to divide the index set $\{0, \ldots, n\}$ into intervals of constant past dimension and then make arguments about the locations $\{R_i\}$. To be explicit, in this notation, $\dim S_{p_i}^\perp < \dim S_{p_{i+1}}^\perp$ and $\dim S_{p_i}^\perp = \dim S_{p_{k+1}}^\perp$ if and only if $R_a < i < R_{a+1}$. The proof is purely algebraic.

**Proposition 6:** Let unprimed quantities be with respect to the syndrome trellis and primed quantities be with respect to any other trellis for the same code. Denote by
\[ \mathcal{E}_j = \sum_{v \in V_j} (\deg_{\text{out}}(v) - 1) \]
the number of expansions at depth $j$ and likewise for the primes, and define
\[ \Delta_j := \sum_{j=0}^{i-1} p^{\dim S_{p_i}^\perp} (\mathcal{E}_j - \mathcal{E}_j). \]

Then, for $R_\kappa < i \leq R_{\kappa+1}$,
\[ \sum_{j=0}^{i-1} p^{\dim S_{p_i}^\perp} (\mathcal{E}_j' - \mathcal{E}_j) = \Delta_j + \sum_{a=1}^{\kappa} p_{\kappa,a}(p_a - 1) \Delta_{R_a} \]
where $p_a = \dim S_{p_{R_a}}^\perp - \dim S_{p_{R_{a-1}}}^\perp$ and $p_{\kappa,a} = \dim S_{p_{R_a}}^\perp - \dim S_{p_{R_{a-1}}}^\perp$.

**Theorem 4 (See [63, Th. 6]):** Let $|E|$ and $|V|$ be with respect to the syndrome trellis and $|E'|$ and $|V'|$ be with respect to any other definition of a trellis for the same code. Then,
\[ |E'| - |V'| + 1 \geq |E| - |V| + 1. \]
Since the syndrome trellis is a minimal representation of the code that is invariant with respect to the generators, following Theorems 3 and 4, it is often argued in the classical literature that the quantity \( |E| \) should be regarded as a fundamental description of how hard it is to decode a given code, rivaling in importance with \( n, k, \) and \( d \). Adopting this philosophy shows, for example, that the color codes are fundamentally more difficult to decode than the rotated surface code of the same distance without invoking projections or hypergraph matching. We provide quantitative data on this in Example 7 in the next section after introducing the CSS splitting of trellises.

Proposition 1 shows that the trellis essentially functions as a compact lookup table for \( S^\perp \). The key to the efficiency of the Viterbi algorithm is its ability to make decisions about all of the elements of this set without computing every path individually. For a given \( v \in V_i \), edge sharing in the trellis enables the algorithm to simultaneously check every element in \( S_i^\perp \) ending at \( v \). When it discards all outgoing edges for \( v \), a total of \( |S_i^\perp| \) elements are eliminated from \( S_i^\perp \) for further consideration (see Fig. 3).

Assuming that \( |E| \) scales at least cubically in \( n \), the cost of the Viterbi algorithm also dominates that of finding the pure error \( T \) given the syndrome. Finding a pure error with the given syndrome is not difficult. The trellis decoder may use any valid \( T \) since the minimum-weight solution is an element of the set \( LST^\perp \) enumerated by the paths of the trellis and will, hence, be found by the Viterbi algorithm. Given a potentially high-weight pure error, decoding may, therefore, be interpreted as a refinement process to the minimum-weight solution. Here, we use pseudoinverse for the syndromes to find \( T \).

**IV. CSS CODES**

CSS codes have the property that the generators of \( S (S^\perp) \) split into those with only powers of \( X \) or only powers of \( Z \). It is common in quantum error correction to decode each set of generators independently and then combine the results into a single correction. This has the advantage of reducing decoding complexity and enabling parallelization at the expense of ignoring potential X–Z correlations. The same technique can, of course, be used with trellises. As a first example, consider the trellis diagrams in Appendix B. Fig. 9(c) shows the trellis diagram of the distance three rotated surface code, and Fig. 9(a) and (b) shows the effect on the trellis of considering the \( X \) and \( Z \) stabilizers separately. Fig. 10 shows the same for the distance three color code. The reduction in trellis and decoding complexity is immediately apparent even for these small examples.

An immediate consequence of decoding the \( X \) and \( Z \) stabilizers separately is that moving from left-to-right in the TOF, the past (future) can only increase (decrease) by a maximum of one stabilizer generator: \( \dim S_{i-1}^\perp \leq \dim S_{i+1}^\perp \leq 1 \) and \( \dim S_i^\perp - \dim S_{i+1}^\perp \leq 1 \). Thus, in contrast to Corollary 2, none of the more complicated edge configurations in Table 1 with a two are allowed, and the edge configurations with the highest contribution of edges do not occur, i.e.,

\[
|V_X/Z| - 1 = 1 \cdot (\frac{3}{2}) + 1 \cdot (\frac{1}{2})
+ p \cdot (\frac{3}{2}) + 1 \cdot (\frac{1}{2})
+ p^2 \cdot (\frac{1}{2}).
\]

and

\[
|E_X/Z| = 1 \cdot (\frac{3}{2}) + p \cdot (\frac{3}{2})
+ p \cdot (\frac{1}{2}) + p^2 \cdot (\frac{1}{2}).
\]

Thus, stabilizer CSS codes that have unit dimension change in the TOF of the \( X \) and \( Z \) stabilizers at the same index \( i \) will have higher dimension changes and hence more edges and will, therefore, be more difficult to decode than those that do not. It follows that self-dual codes are more difficult to decode than non-self-dual codes of the same parameters. Note that applying \( E = M \) to CSS trellises gives [64]

\[
\#(\frac{3}{2}) = \#(\frac{1}{2}).
\]

Let \( S \) be a stabilizer code given by the CSS construction with \( C_1 = [n, k_1, d_1]_\rho \), \( C_2 = [n, k_2, d_2]_\rho \), and \( C_2^\perp \subseteq C_1 \). Denote by \( G_i \) and \( H_i \) the generator and parity-check matrices for \( C_i \), respectively. The stabilizers of the quantum code are given, in symplectic form \((X | Z)\), by \( H_1 \oplus H_2 \). The set \( S^\perp \) is generated by the corresponding normalizer matrix [65], [66], [67]

\[
\begin{pmatrix}
0 & C_1 \\
C_2 & 0
\end{pmatrix}
\]

The trellis for the \( X \) (\( Z \)) stabilizers only in a CSS code is, therefore, determined by the classical trellis whose paths are in one-to-one correspondence with the codewords of \( G_1 \) (\( G_2 \)). Edge labels for CSS trellises should be restricted to \( I \) and \( Z \) for \( I \) and \( X \) with weights \(-\log \Pr (I) - \log \Pr (X)\) and \(-\log \Pr (Y) - \log \Pr (Z)\) for \( X \) and similarly for the \( Z \) stabilizers.

It is well known in classical trellis theory that the dimension of \( V_i \) is equal to the dimension of the corresponding \( V_i \) for the dual code. Since one has \( p^{n-1} \) possible syndromes and the other \( p^k \), we immediately get the following.

**Corollary 6 (Wolf Bound for CSS Codes [53]):**

Let \( C_i \) be as above. Then, for the \( X \) stabilizer trellis,
\[ |V_{X,i}| \leq p^{\min\{k_1, n-k_1\}} \], and for the Z stabilizer trellis, \[ |V_{Z,i}| \leq p^{\min\{k_2, n-k_2\}} \].

Example 5: The distance three color code is equivalent to the well-known [[7, 1, 3]] Steane code. This is a CSS code constructed with the [7, 4, 3] binary Hamming code and its dual. Hence, \[ |V_{X/Z,i}| \leq 2^{\min\{3,4\}} = 8 \], which agrees with Fig. 10.

It is perhaps not surprising that the full trellis for a CSS code turns out to be a product of the two trellises of its component codes. For instance, the active generators of the full code are the active generators of the X code and the active generators of the Z code. Thus

\[ |V_i| = p^{\dim S_{X,a} + \dim S_{Z,a}} = p^{\dim S_{X,a} p^{\dim S_{Z,a}}} = |V_{X,i}||V_{Z,i}|. \]

Definition 10 (Trellis PRODUCT [61]): The trellis product of two trellises with vertices \( V \) and \( V' \) and edges \( E, E' \), respectively, is denoted by \( \square \) and has vertices

\[ V \square V' = \bigcup_{i=0}^{n} V_i \times V'_i \]

and edges

\[ E \square E' = \{ ((v_i, v'_i), PP', (v_{i+1}, v'_{i+1})) | (v_i, P, v_{i+1}) \in E_i, (v'_i, P', v'_{i+1}) \in E'_i \}. \]

Remark: This is sometimes called the Shannon product [68] for historical reasons after Claude Shannon who first described the product of two channels operating at the same time or also the tensor product [69]. However, much of the classical literature on the trellis product is based on nonsyndrome trellises and does not apply here.

Example 6: Using shorthand \( \Delta \dim S_{X,i}^{1/2} = \dim S_{X,i}^{1/2} \) and \( \Delta \dim S_{Z,i}^{1/2} = \dim S_{Z,i}^{1/2} \) for edge configurations in Table 1, it may be checked that \( (m, n) \times (m', n') = (m + m', n + n') \). Such configurations occur in the full trellis as graph products of lower-dimensional configurations in the CSS trellises. The vertices of Fig. 9(c) are ordered by the trellis product of Fig. 9(a) by Fig. 9(b) and the vertices of Fig. 10(b) by the trellis product of Fig. 10(a) with itself.

Lemma 5: Let \( S \) be a CSS code with X stabilizers, \( S_X \), and Z stabilizers, \( S_Z \). Let \( V_X, E_X \), \( \deg_{\text{in},X} \), and \( \deg_{\text{out},X} \) denote the appropriate quantities for the trellis of \( S_X \), and likewise for \( S_Z \). Then, the trellis for \( S \) is given by the trellis product of the trellises for \( S_X \) and \( S_Z \). Furthermore, we have the following.

i) \[ |V_i| = |V_{X,i}||V_{Z,i}|. \]

ii) \[ |E_i| = |E_{X,i}||E_{Z,i}|. \]

iii) \[ \deg_{\text{in},i} = |\deg_{\text{in},X,i}||\deg_{\text{in},Z,i}|. \]

iv) \[ \deg_{\text{out},i} = |\deg_{\text{out},X,i}||\deg_{\text{out},Z,i}|. \]

The proof of the enumerated parts of this lemma follows from Definition 10, and the larger claim is a restatement of the well-known fact that \( X \) and \( Z \) errors may be decoded independently for CSS codes. A rigorous proof is fairly trivial, and we leave it to the reader. These results may be demonstrated with the diagrams in Appendix B. Note that the space theorems in Theorem 1 do not hold for CSS splittings. Following the proof of the theorem, the number of vertices is isomorphic to \( |S^{1/2}/\ker \sigma_i| \) but now the kernel with respect to \( S_X \) includes \( S_X^{1/2} \), as well as the past and future elements of \( \delta_{Z,i}^2 \):

\[ |V_{X,i}| = p^{\dim S_{X,i}^{1/2} - \dim S_{Z,i}^{1/2} - \dim S_{Z,i}^2 - \dim S_{Z,i}^4}. \]

Similar equations hold for \( |V_{Z,i}| \) and the edges. The same idea holds for counting vertices and edges for an arbitrary subset of generators of \( S \). This split trellis idea may be used in more generality. The proof of the following with the clear associativity of the trellis product implies the main part of the lemma above.

Theorem 5: For \( S = \{ s_1, \ldots, s_{n-k} \} \), the minimal trellis for \( S \) is given by the trellis product of the minimal trellises for each \( s_i \).

This splitting idea may be used to decode in a similar manner to the CSS codes where each grouping of syndrome bits is corrected independently and then combined. The success of this approach is tied to finding an appropriate grouping of generators with respect to which the logical operators split "nicely" such that the combined corrections do not introduce a logical error not returned by the individual trellises.

If we adopt the idea that \( |E| \) represents a fundamental description of how difficult it is to decode, then the two quantities of interest to compare are \( |E_X| + |E_Z| \) and \( |E| \), where the latter is with respect to the full stabilizer code. In particular, for a self-dual code, \( 2|E_{X/Z}| \leq |E_{X/Z}|^2 \). It follows that splitting the decoding of a self-dual CSS code is square-root easier than decoding the full code. A rather trivial bound on the more general case is immediate.

Proposition 7: Let \( |E| \) be the total number of edges in a trellis diagram for a CSS code whose \( X \) and \( Z \) codes have trellises with a total number of edges \( |E_X| \) and \( |E_Z| \), respectively. Then

\[ |E_X| + |E_Z| \leq |E| \leq |E_X||E_Z| - p^2 n (n - 1). \] (19)

The right-hand side of (19) is tight for the case that there are a minimal number of edges at each depth such that \( |E_i| = 4 \) for all \( i \). This, of course, is rare, usually only occurring around the left and right ends of the trellis, and the more the trellis deviates from this value, the worse this upper bound becomes.

It would be more useful to obtain bounds on the number of edges in any of the diagrams with only the values \( n, k, \) and \( d \). We leave this for future work and simply note that several classical trellis bounds could potentially be exploited for CSS codes. An easy one in particular worth mentioning is the case of classical maximum distance separable (MDS) codes where it is well known that the sequence \( \{||V_i|\| \} \) has the following pattern [53]:

\[ \{1, p, p^2, \ldots, p^{\min\{k,n-k\}}, p^{\min\{k,n-k\}}, \ldots, p^2, p, 1\}. \]
Combining this with the fact that \( E_i \geq V_i \), removing the contribution from \( V_0 \), and summing the geometric series, we get

\[
|E_{X/Z}| \geq 2^{p \min\{k,n-k\}} - p - 1
\]

which is tight for the (unrealistic) case that the outgoing degree at every \( i \) is one. Replacing \( p \) with \( p^2 \) in (20) provides the corresponding bound on \( |E| \) for a self-dual CSS code. Combining the two loose bounds (19) and (20) does not appear useful.

**Example 7:** Returning to the proposal that \( |E| \) represents a fundamental parameter for the code, vertex and edge counts for the 4.8.8 and 6.6.6 color codes, rotated surface codes, and their CSS splittings are given in the following tables. The non-CSS XZZX surface codes have the same values as the full rotated surface codes. A standard qubit numbering order was applied to the surface codes, but the qubit numbering for the color codes was assigned greedily to minimize the trellis. The difference in values between distances may become more consistent given more consistent numbering schemes. Note that the total vertex and edge counts for the CSS trellises are the sum of the \( X \) and \( Z \) counts.

The difference in sizes between the \( X \) and \( Z \) trellises for the rotated surface code family is an artifact of the particular numbering system used in this work. Consider the \( X_2X_3 \) stabilizer in Fig. 4(a). This is active only at depth 2 since at depth 3 its syndrome is already zero. On the other hand, the \( Z_1Z_6 \) stabilizer is active over five depths and, hence, contributes vertices over this entire range. Since the \( X \) and \( Z \) stabilizer measurements are independent, arranging the data with respect to two different numbering schemes will allow both the \( X \) and \( Z \) trellises to be isomorphic. This demonstrates the strong effect permutations have on the trellis.

While an analytical formula for the edge scaling is currently missing, we can gain some numerical insights from Table 3. In Fig. 4(b), we see that \( |E| \) is exponential in the minimum distance \( d \) [compare to the lines \( y = 2d \) (blue) and \( y = d \) (orange)]. The slight bend in the data points, most predominately seen for the CSS rotated surface codes, suggests a more complicated behavior. It is possible that the color codes also have the same trend but were not simulated to high enough distances to make this as visually apparent. It is also possible that the greedy algorithm used to assign qubits to the color code geometries was suboptimal enough to erase this effect.

**Example 8:** To further emphasize the effect permutations have on the trellis, draw the distance five and distance seven 4.8.8 color codes, choose an arbitrary boundary, and number the qubits from left to right, level by level. The minimum trellis for this configuration for the distance five code has \(|V| = 5242\) and \(|E| = 9000\) and has \(|V| = 177,018\) and \(|E| = 293,928\) for distance seven. Compared to Tables 3 and 4, using this numbering scheme would increase the slope of the blue dots in Fig. 4(b), potentially limiting the ability to do large-scale simulations at a lower distance.

**V. SIMULATIONS AND DISCUSSION**

Numerical simulations were performed in the Julia programming language with presimulation computations in the MAGMA quantum coding theory library [70]. We begin by
TABLE 4. Edge Counts by Distance for Common Codes

| Code   | 3    | 5    | 7    | 9    | 11   | 13   | 15   | 17   | 19   | 21   |
|--------|------|------|------|------|------|------|------|------|------|------|
| 4.8.8  | 232  | 7080 | 14327| 3559336| 14506536| 175628968| 335695592| 4391000| 2020188| 1007967784| 140007320456| 60870993576| 1031533604008| 1742112880544|
| 4.8.8 X/Z | 36   | 316  | 1884 | 11100| 27884| 89628| 4391000| 2020188| 60870993576| 1031533604008| 1742112880544|
| 6.6.6  | 232  | 4648 | 89512| 832936| 7708072| 89669032| 1007967784| 733820456| 64652391976| 2316044| 791674856| 3934257128|
| 6.6.6 X/Z | 36   | 236  | 1340 | 5372 | 19388| 65852| 234956| 82856| 2316044| 791674856| 3934257128|
| RSsurf | 152  | 2152 | 19688| 143336| 913384| 5341160| 29425640| 155189224| 791674856| 3934257128|
| RSsurf X | 30   | 172  | 700  | 2388 | 7316| 20832| 56436| 146932| 371188| 915444|
| RSsurf Z | 44   | 284  | 1228 | 4332 | 13548| 39148| 106988| 280556| 712084| 1765356|

constructing the trellis. Historically, it is widely assumed, even to this day, that the trellises considered in this work are too large to ever be used. Fortunately, the commonly referred to size estimates date back to the late 1980s, early 1990s when constructing the multigigabyte trellises required for color codes of distance 21 would have indeed been impossible.

The construction algorithms in [41] and [44] iterate through all $p^{n+k}$ elements of $S^\perp$. The theoretical guarantees of the space theorems in Theorem 1 allow us to do better by only generating elements until we hit $|V|$ and $|E|$. Numerical trials using this method suggested that the largest trellis considered in this work would take between one to two months to generate. Many classical construction algorithms are just as wasteful [53].

Our algorithm is closest to [71] for classical codes, but is conceptually simpler and more efficient. The basic idea is to serially proceed through $V_i$ picking a vertex and examining all possible outgoing edges for matches in $V_{i+1}$. This implicitly utilizes Corollary 1, although this result never appears in the literature. Here, we rely on the theoretical guarantees provided by Corollary 1, Theorem 1, Lemma 1, and Theorem 2. By Theorem 1, the vertices at depth $i$ are given by the set of all $(n - k)$-tuples whose bits corresponding to generators active at $i$ range through all possible values. To construct the edges at section $i$, determine all of the vertices in $V_{i-1}$ connecting to $\overline{e}_i$ via Corollary 1. Then, choose one of these $v \in V_{i-1}$ and determine all of the vertices in $V_i$ to which it connects, again via Corollary 1. These are all of the vertices in the edge configuration by Theorem 2, which may be completed with Corollary 1. This is the subgroup of $E_i$ by Lemma 1. The rest of the section consists of all translations (cosets) of this subgroup. Our new proof of the classification of edge configurations allows us to only compute a small bipartite graph for each depth to generate the entire trellis. $V_i$ are independent and may be constructed in parallel, after which $E_i$ may also be parallelized. For the codes considered in this work, only the distance 19 and 21 color code trellises were large enough to justify saving so as not to compute them on-the-fly later.

As previously mentioned, the vertex labels are required for the construction of and theoretical justification for the trellis, but an examination of the Viterbi algorithm shows that they serve no role in decoding. As such, we need not bother shifting the vertex labels for each measured syndrome, and generally, we can safely discard them after the construction phase is completed. Even storing the labels as integers is similar to creating a lookup table and can take a nontrivial amount of memory starting at moderately sized codes. The storage requirements for the zero-syndrome trellis valid for any error model became slightly unwieldy to use for large-scale simulations for the distance-21 color codes, but this was implementation dependent. Choosing a specific error channel allowed the trellis to be translated to a data structure a fraction of the storage size of the original. For example, the distance-21 4.8.8 CSS trellises come out to around 300 MB in contrast to the roughly 2 GB file size of the original data structure, which includes all the information and generality. We leave a discussion of such improvements to the next chapter. However, since the trellis searches all elements of $S^\perp$, the proper comparison to make here is to the size of a lookup table for the same code. Assuming that the $2^{n+k}$ elements of $S^\perp$ are stored as length-$n$ strings of character size 1 byte, a lookup table for the $[[111, 1, 11]]$ code above would be on the order of $\sim 10^{17}$ exabytes.

To exemplify the versatility of trellis decoding, the row $n = 20$ was selected at random from the quantum error-correcting codes table at [51]. Extended codes (with weight one stabilizers) and codes with $k = 0$ or $d \leq 2$ were discarded, and code-capacity (memory model) numerical simulations were performed on the remaining four codes: $[[20, 3, 6]], [[20, 4, 6]], [[20, 10, 4]],$ and $[[20, 13, 3]]$. These codes and their properties were previously unknown to the authors of this work and were never investigated. In particular, it is unknown if another decoding method is known for these codes and if there exist previous pseudothreshold results in the literature. Simulations consisted of a single round of error correction under a depolarizing noise model acting independently on each qubit. Error correction was considered to have failed if the final state contains a logical error with respect to any of the $2k$ logical operator generators. Results are presented in Fig. 5, and the stabilizers for these codes are given in Appendix C for convenience.

In order to quantitatively assess the success of trellis decoding, the same simulations were carried out for three common qubit code families, the rotated surface and 4.8.8 and 6.6.6 color codes for $Z$ only noise

$$E(\rho) = (1 - p)\rho + pZ\rho Z$$

a single-axis dephasing channel (using the CSS $X$ only trellis) up to at least distance 17. These codes and this noise model were chosen solely for the purpose of comparison with existing decoding literature. Due to the $X-Z$ symmetry of the
codes, the same results are obtained for independent X channel and depolarizing thresholds are at 3/2 times the X or Z noise results decoded independently, which we numerically verified for low distances. There are, of course, many other highly optimized decoders for topological codes, and these choices may not sufficiently demonstrate the value of trellis decoding, which shines for moderately sized quantum error correcting codes (QECCs) for which there are no efficient decoding strategies. Data were taken at intervals of size 0.0025 around the suspected threshold to reduce the error caused by sampling and fitting.

Previous studies have reported a surface code Z threshold of 10.3% under MWPM [72] and 10.9% using the tensor network decoder of [27]. The statistical mechanical threshold is estimated to be 10.9% [73] without correlations and 12.6% with X and Z correlations taken into account [74]. Both the trellis and MWPM are minimum-weight decoders for this channel, so they should agree for uncorrelated noise. For correlated error models, trellis decoding should beat MWPM since the minimum-weight correction on the full trellis will not always match the minimum-weight X correction combined with the minimum-weight Z correction returned by MWPM. For example, consider the error $Y_8Y_{13}Y_{18}$ on the distance-5 rotated surface code of Fig. 4(a). The full trellis will return the correction $Y_8Y_{13}Y_{18}$, while the X- and Z-trellises will return $Z_3Z_{23}$ and $X_8X_{13}X_{18}$, respectively, the latter of which is a logical error. The Viterbi algorithm takes the same number of steps every time and could be slower than using a small matching graph for a large code, so the expected number of errors should be taken into account when comparing the runtime of these two algorithms for uncorrelated error models. The results of our simulations are shown in Fig. 6(a).

The restriction decoder of [11] achieves a Z threshold of 10.2% for the 4.8.8 color codes with periodic boundary conditions. This was modified to include boundaries in [75], which reports a full depolarizing noise threshold of 12.6% for the 6.6.6 lattice, which results in a Z threshold of roughly 8.4%. Bombin et al. [7] find a Z threshold of roughly 8.7% for the 4.8.8 lattice. The statistical mechanical thresholds for both lattices are estimated to be 12.6% and 12.52% for the 6.6.6 and 4.8.8 families, respectively [74], [76]. The results of our simulations for the 6.6.6 and 4.8.8 families are shown in Fig. 6(c) and (e), respectively.

Following studies such as [4], we perform a finite-size threshold analysis on our data by fitting to the ansatz

$$A + B(p - p_{th})d^{1/v} + C(p - p_{th})^2 d^{2/v}$$

for $A, B, C, p_{th}$, and $v$ starting with $d \geq 9$. For the rotated surface codes, we find $p_{th} \approx 10.25\%$ ($v = 1.58$). It is noteworthy that Fowler et al. [77] did not see convergence to this value below $d = 21$. For the color codes, it is possible that all of the distances reported in this work are within the finite-size-effect regime. Nevertheless, we see strong evidence for threshold-like behavior at 10.1% ($v = 1.29$) and 10.4% ($v = 1.51$) for the 6.6.6 and 4.8.8 families, respectively. Further simulations and analysis are required to better estimate the thresholds for these codes using this decoder. Fig. 6(b), (d), and (f) shows the logical error rate versus the rescaled Z noise probability $x = (p - p_{th})d^{1/v}$ for data near each threshold, along with the line of best fit.

Judging by the time it took to complete each simulation, we believe that the CSS distance-21 trellis for the 4.8.8 color code ([421, 1, 211]) roughly represents the limit for large-scale simulations. Considering $|E|$ as a measure of the difficulty of decoding, trellis profiles for codes were compared to Tables 3 and 4 before using this method. We offer this as a rough benchmark but acknowledge that some of this may due in part to not properly leveraging the limited computational resources available for this work. For example, we noticed that a large percentage of the total simulation time for the distance-19 and distance-21 color codes was consumed by distributing the trellis to all the processors. We have chosen to use this chapter as a baseline for trellis theory, including its limits. Reporting all improvements to the theory or its practical implementation has been relegated to the next chapter. The special structure of some codes may be utilized to reduce $|E|$ without the need for further theory, as demonstrated by the next example.

**Example 9:** Let $H$ denote the numerical matrix representing the X or Z stabilizers for the [[7,1,3]] Steane code

$$H = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$  

The level-2 concatenated Steane code is the [[49, 1, 9]] code whose X or Z stabilizers are given by the Pauli operators corresponding to $I \otimes H$ and $H \otimes \tilde{I}$, where $I$ is the $7 \times 7$ identity matrix and $\tilde{I}$ is the length-7 all-ones vector. The trellis for this code has $|V| = 626$ and $|E| = 844$. Using Theorem 5 and the subsequent discussion, we consider a two-stage suboptimal decoder. The seven copies of $H$ are decoded independently at a cost of 36 edges each. The resulting corrections are made; then, the syndrome updated and the bits corresponding to

\[\text{FIGURE 5. Simulated logical error rates for the four codes in the row } n = 20 \text{ satisfying our requirements at [51]. Triangular data points are importance sampled to a tolerance of } 10^{-3} \text{ and circular points are direct sampled; both methods were used and were found to agree at } p = 0.01. \text{ The black line is } y = x.\]
$H \otimes \bar{I}$ are again decoded using the trellis for $H$. The correction returned from this trellis is tensored with $\bar{I}$ to map back to the original problem. Since the first seven decodes can be done simultaneously, the time complexity of this decoder is determined by only $36 + 36 = 72$ edges, a 91.5% savings. Numerical simulations similar to the ones above were performed for both this decoder and the normal single-stage trellis. Results are presented in Fig. 7, where we see that this is indeed suboptimal trading speed for accuracy.

The calculated pseudothreshold for the independent $Z$ channel between the Steane code and concatenated Steane code using two-stage block decoding is 6.44%, in agreement with the $Z$ threshold of 6.46% estimated from the depolarizing channel threshold of 9.69% [78]. The calculated $Z$ pseudothreshold between the Steane and the concatenated Steane code using the full trellis is 10.55%. This is known to be below the true threshold for continued concatenation based on message passing of 12.5% estimated from a depolarizing channel threshold of 18.8% [79]. The $[[7, 1, 3]]$ Steane code can also be considered a distance-3 6.6.6 color code. It is informative to compare the performance of the $[[61, 1, 9]]$ 6.6.6 color code to the level-2
concatenated Steane code. We see that the color code has a slightly better performance at the cost of 12 more qubits.

The numerical threshold estimates obtained in this work are competitive with, and sometimes even better than, current state-of-the-art decoders. We attribute this success to two main things. First, with the exception of organizing the data into logical cosets, the decoder is provided with close to the maximum amount of information possible of both the code and the error channel. Traditional decoders often only rely on a fraction of this information. Second, since every element of $P_n$ represented by a path in the trellis has the same syndrome as was measured, any correction returned by the decoder will, at minimum, force the resulting state to have zero syndrome. If there are few or no logical operators of the weight of the true error, the decoder will, therefore, correct the majority of the correctable errors of that weight regardless of the minimum distance of the code.

VI. CONCLUSION
We hope to reinitiate interest in trellis decoding for stabilizer codes with this work. Starting from the basic construction in [41], we have provided a theoretical foundation required for any future analysis. In doing so, we have detailed the use of, improved the runtime of, and demonstrated the effectiveness of trellis decoders. The qualitative metric we propose shed light on the difference between color codes and surface codes and CSS and non-CSS stabilizer codes. In particular, the trellis is a native color code decoder without invoking projections, restrictions, homology, charges, excitations, string nets, or boundaries.

The theoretical work here attempted to closely parallel classical trellis theory, yet also introduced new results unique to the quantum setting. Comparing this work with the remaining classical literature, a considerable number of open problems remain. In particular, many, if not most, fundamental results of the classical theory rely on the relationship between a code and its dual. The concept of duality is a bit trickier for stabilizer codes, and we address it a follow-up work.

An important set of classical results missing quantum analog are bounds for various quantities given only $n$, $k$, and $d$ [55], [60], [64], [80], [81], [82], [83], [84]. This is easier in the classical setting as the minimum distance is related to the number of paths between a vertex in $V_i$ and a vertex in $V_{i+j}$ for some $j$. Here, this would be the minimum distance of $S^\perp$ and not $S^\perp \setminus S$. A thorough analysis will also shed light on the critical open question of the scalability of trellis decoding, which is required to properly compare it with other decoding techniques. Having a complete answer to this question will allow for the analysis of the asymptotic trellis decoding behavior of various code families. The question then becomes at what point this scaling becomes too cumbersome for modern quantum controller hardware. In such cases, trellis pruning [85] or coset techniques may possibly be employed. Classically, trellis decoding becomes computationally more intensive as the encoding rate increases, since $\max_i |V_i|$ grows with $k$.

While the code-capacity results are promising, further numerical studies are needed to determine the behavior of this method under the more realistic and interesting phenomenological, circuit-level, and biased noise models. Incorporation into other paradigms, such as flag-syndrome extraction and single-shot error correction, are also interesting. Trellises are a hidden Markov model for the decoding process, and it is intriguing to consider whether a Markov chain with memory could be used to handle correlated error models. Qudit codes that were not investigated during this work are important to investigate as decoders for such codes are fewer in number and are not as well understood as their qubit counterparts. The treatment of CSS codes in this work may lead to further quantitative results for these codes. Along these lines, it is common in classical coding theory to exploit the coset structure of codes, which make a visual appearance in the trellis [86], [87]. The study of such trellis substructures may provide new insight into the structure of stabilizer codes.
VITERBI ALGORITHM

FIGURE 8. Viterbi algorithm applied to the example trellis of Fig. 1 for the error model $\Pr(I) = 0$, $\Pr(X) = \Pr(Z) = 1$, and $\Pr(Y) = 2$. Ties were broken in a manner to keep the result looking clean. The path from $V_0$ to $V_n$ provides the final correction, in this case $III/III$—which is true since there was no error. (a) Section 1. (b) Section 2. (c) Section 3. (d) Section 4. (e) Section 5.
Algorithm 1: Viterbi.

**Input:** A vertex set, $V$, along with an edge set, $E$, corresponding to a valid trellis.

**Output:** A Pauli string

1. Define a `Struct` `Vertex` that contains:
   - `int prev` → Final step acts as a linked list
   - `float value` → Initialized to null
   - `Edge edge`

2. Define a `Struct` `Edge` that contains:
   - `Vertex source`
   - `float weight`
   - `char label`

3. Let $V_0 \ni \bar{0}.value \leftarrow 0$

4. **for** $i \leftarrow 1 \text{ to } n$
   
   ```
   foreach v \in V_i do
   v.value \leftarrow \min \{e.source.value + e.weight\} \hspace{1em} \text{For } e \text{ which achieves minimum}
   v.prev \leftarrow e.source
   v.edge \leftarrow e
   end
   ```

5. Let $\bar{V}_n \ni 0 \in V_n$

6. **for** $i \leftarrow n \text{ to } 2 \text{ by } -1$
   
   ```
   correction \leftarrow v.edge.label + correction
   v \leftarrow v.prev
   ```

7. **return** `correction`
APPENDIX B
DISTANCE-3 ROTATED SURFACE AND COLOR CODE DIAGRAMS

**FIGURE 9.** Trellis diagrams for the distance three rotated surface codes. (a) X-stabilizers only. (b) Z-stabilizers only. (c) Full code with vertices organized by the trellis product of (a) with (b). It is common to collapse edge sections when $\text{deg}_{\text{out}} = 1$, combining labels with the section before it. This is called **sectionalization** and will not be discussed further in this work. It is known that sectionalization can reduce $\max |V|$ but not $\max |E|$ [60].
FIGURE 10. Trellis diagrams for the distance three color code. (a) X-stabilizers (or Z-stabilizers) only. (b) Full code with vertices organized by the trellis product of (a) with itself.
APPENDIX C
CODES OF FIG. 5

**APPENDIX D**
PROOFS OF TECHNICAL LEMMAS

**Proposition 1:** There is a one-to-one correspondence between Pauli strings in $S^\perp$ and length-$n$ paths in the trellis.

**Proof:** It is clear that by construction every Pauli string in $S^\perp$ corresponds to a length-$n$ path in the trellis. It remains to show that every length-$n$ path in the trellis corresponds to a Pauli string in $S^\perp$ and that all such paths are unique.

Let $(\sigma_1, P_1, \ldots, P_n) \in E_i$ be an edge with label $P_i$; then, pick an arbitrary edge $(\sigma_1, P_1, \ldots, P_n, \ldots, P_{n-1}, \ldots, P_n) \in E_j$. Continuing this process, we end with an edge $(\sigma_{n-1}, P_{n-1}, \ldots, P_0) \in E_k$. This last term is the zero syndrome by construction; hence, concatenating the edge labels in the path, the Pauli string $P_{n+1} P_{n} \ldots P_0$ has length-$n$ and zero syndrome and is, hence, an element of $S^\perp$. The result then follows from the previous proposition.

**Corollary 1:** Let $v_i \in V_i$ and $v_{i+1} \in V_{i+1}$ be arbitrary. If there is a Pauli element, $P_i$, which takes $v_i$ to $v_{i+1}$, then $(v_i, P_i, v_{i+1}) \in E_i$.

**Proof:** Let $v_i \in V_i$ and $v_{i+1} \in V_{i+1}$, and suppose that concatenating a Pauli element $P_i$ to a path with terminus $v_i$ changes the syndrome of $v_i$ to the syndrome of $v_{i+1}$. Let $V_0 v_i$ and $v_{i+1} V_n$ be a path from $V_0$ to $v_i$ and $v_{i+1}$ to $V_n$, respectively. Then, the Pauli string represented by the path $V_0 v_i P_i v_{i+1} V_n$ has a length-$n$ and zero syndrome and is, hence, an element of $S^\perp$. The result then follows from the previous proposition.

**Proposition 2 (See [41, Lemma 1]):** Given a stabilizer code $S$, any trellis for $S$ must satisfy

$$|V_i| \geq p^{\dim S^\perp - \dim S_{p_i}^\perp - \dim S_{i}^\perp}$$

$$|E_i| \geq p^{\dim S^\perp - \dim S_{p_{i-1}}^\perp - \dim S_{i}^\perp}.$$
Proof: Every path from $V_0$ to $V_n$ must go through some vertex $v \in V_i$. The set of all Pauli strings in $S_i$, which map to the vertex $v$ under $\sigma_i$, is a subset of the cosets described in Fig. 3 and, hence, has cardinality bounded above by $|S_{P_i}| |S_{I_i}^⊥|$. The trellis can be written as a union of paths passing through $v$ over all $v \in V_i$, so the number of such cosets is $|V_i|$. Using Proposition 1, we then have

$$|S_i^⊥| \leq |V_i| |S_{P_i}^⊥| |S_{I_i}^⊥|. $$

Likewise, every path in the trellis contains one edge in section $E_i$. Since edges are of the form $(s(e), P, t(e))$, a similar argument gives

$$|S_i^⊥| \leq |S_{P_i}^⊥| |E_i| |S_{I_i}^⊥|. $$

**Theorem 1 (Quantum SPACE THEOREMS):** The syndrome trellis of Section II is minimal, i.e.,

1) (quantum state-space theorem; see [41, Lemma 2])

$$|V_i| = P^{|\text{dim} S_i^⊥ - \text{dim} S_{P_i}^⊥ - \text{dim} S_{I_i}^⊥|}.$$  

2) (quantum branch space theorem)

$$|E_i| = P^{|\text{dim} S_{P_i}^⊥ - \text{dim} S_{I_i}^⊥|}.$$ 

**Proof:** Let $i$ be fixed.

1) The vertices are the image of $S_i^⊥$ under $\sigma_i$. By the above comments, the past and the future have zero syndrome; therefore, $V_i = \sigma_i(S_i^⊥) \cong S_i^⊥/S_{P_i}^⊥ \times S_{I_i}^⊥$ by the first isomorphism theorem for groups.

2) The kernel of the map from $S_i^⊥$ to $E_i$ consists of the Pauli strings, which map to $(\emptyset, I, \emptyset)$, i.e.,

$$\ker \sigma_{i-1}(S_i^⊥) \cap \{P \in S_i^⊥ \mid P = I\} \cap \ker \sigma_i(S_i^⊥)$$

isomorphic to $S_{P_{i-1}}^⊥ \times S_{I_{i-1}}^⊥ \cap \{P \in S_i^⊥ \mid P = I\} \cap S_{P_i}^⊥ \times S_{I_i}^⊥$.

$$= S_{P_{i-1}}^⊥ \times S_{I_{i-1}}^⊥ \cap \{P \in S_i^⊥ \mid P = I\} \cap S_{P_i}^⊥ \times S_{I_i}^⊥$$

Lemma 1: The set of edges in $E_i$ with terminus vertex $\emptyset$ is a subgroup of $E_i$.

**Proof:** The proof of Theorem 1 shows that $E_i$ is a group. Clearly, the identity edge exists and closure follows since if $(s_1, P_1, \emptyset), (s_2, P_2, \emptyset) \in E_i$, then $(s_1, P_1, \emptyset)(s_2, P_2, \emptyset) = (s_1 \cdot s_2, P_1 P_2, \emptyset) \in E_i$, where $\cdot$ is the appropriate group (or vector space) binary operation on the vertices. It remains to show that $E_0$ contains inverses. Let $(a, b, \emptyset) \in E_0$ and $(c, d, e)$ be its inverse in $E_i$. The group operation on $E_i$ forces $e = \emptyset$.

**Corollary 2:** For $1 \leq i \leq n$, every vertex $v \in V_i$ has incoming degree

$$\deg_{\text{in}}(v) = P^{|\text{dim} S_{P_i}^⊥ - \text{dim} S_{P_{i-1}}^⊥|}. $$

and for $0 \leq i \leq n - 1$, every vertex $v \in V_i$ has outgoing degree

$$\deg_{\text{out}}(v) = P^{|\text{dim} S_{I_i}^⊥ - \text{dim} S_{I_{i+1}}^⊥|}. $$

**Proof:** Let $v \in V_i$ and denote by $E_{\text{in}}(v)$ the set of edges with terminus $v$, $E_{\text{out}}(v) = \{e \in E_i \mid t(e) = v\}$. By definition, $\deg_{\text{in}}(v) = |E_{\text{in}}(v)|$. By Lemma 1, $E_{\text{in}}(v)$ is a coset of $E_{\text{in}}(\emptyset)$ in $E_i$. In particular, $|E_{\text{in}}(v)| = |E_{\text{in}}(\emptyset)|$. Since there are $|E_i|$ total edges equally divided among $|V_i|$ vertices, $\deg_{\text{in}}(v) = |E_i|/|V_i|$. The result follows from Theorem 1. An identical proof gives the second equality with $\deg_{\text{out}}(v) = |E_{i+1}|/|V_i|$. 

**Lemma 2:** Let $A = S$ or $S^⊥$. Then, $0 \leq \dim A_{i-1} - \dim A_{i-1} \leq 2$ for $0 \leq i \leq n - 1$ and $0 \leq \dim A_{i-1} - \dim A_{i-1} \leq 2$ for $1 \leq i \leq n$.

**Proof:** We show the result for the first inequality and the remaining proof is identical. An easy counting argument shows that the number of $I$s in the $i$th column of $A_{i-1}$ are either $|A_{i-1}|, |A_{i-1}|/p, \text{ or } |A_{i-1}|/p^2$. (If there are not all $I$s, then pair any element with its inverse to create an element with an $I$. Likewise, pair any element without an $I$ with an element with an $I$. Do the same for $X$ and $Z$ combinations.) These are the elements that are also in $A_{i-1}$, so $|A_{i-1}| \geq |A_{i-1}|/p^2$. The dimension change is, therefore, no more than two.

**Theorem 2:** Consider an arbitrary edge $e = (s(e), P, t(e)) \in E_i$ and define $I_{i-1} = \{v \in V_{i-1} \mid \exists e' \in E_{i-1}, t(e') = t(e)\}$ and $I_i = \{v \in V_i \mid \exists e' \in E_i, s(e') = s(e)\}$. Then, the vertices of $I_{i-1}$ and $I_i$ form a completely connected bipartite graph and no other elements of $V_{i-1}/I_{i-1}$ or $V_i/I_i$ are connected to the vertices in $I_i$ and $I_{i-1}$, respectively. If there exists a parallel edge in $E_i$, then all edges in $E_i$ are parallel with the same number of edges in parallel.

**Proof:** From Corollary 1 and Lemma 1, it suffices to show this for the edge $(\emptyset, I, \emptyset) \in E_0$, as all edge configurations are a shift of this one. Choose an arbitrary $v_{i-1} \in I_{i-1}$ and $v_i \in I_i$. By definition, there exists Pauli label $P_i$ and $P_{i-1}$ such that $(v_{i-1}, P_{i-1}, \emptyset), (\emptyset, P_i, v_i) \in E_i$. Hence, $(v_{i-1}, P_i P_{i-1}, v_i) \in E_i$. This works for any pair of vertices in $I_{i-1}$ and $I_i$, proving the first statement. Now, suppose that, without loss of generality, there exists a $v' \in V_i/I_i$ connected to a $v_{i-1} \in I_{i-1}$ via $(v_{i-1}, P', v')$ but is not part of the bipartite graph. As a subgroup, the inverse syndrome to $v_{i-1}$, $v_{i-1}$, exists with some edge label $P_i$. Then, $(v_{i-1}', P_i, \emptyset)(v_{i-1}, P', v') = (\emptyset, P_i P', v') \in E_i$, a contradiction to the fact that $v' \not\in I_i$.

To prove the last statement, pick a source/terminus pair $v_{i-1} \in V_{i-1}$ and $v_i \in V_i$ with parallel edges uniquely labeled by $\{P_1, \ldots, P_k\}$. We will show that every other edge must also have $k$ parallel edges; hence, it suffices, without loss of generality, to assume that both syndromes are $\emptyset$. Now, choose any other edge $(v_{i-1}', P', v') \in E_i$ with $v_{i-1}$ and $v'$ not both $\emptyset$. By Lemma 1, $(\emptyset, P_{i-1}', \emptyset)$ exists and $(v_{i-1}', P', v')(\emptyset, P_{i-1}', \emptyset)(\emptyset, P_i, \emptyset) = (v_{i-1}', P_i P_{i-1}' P_{j', v'}$ for some $1 < j < k$. Since $P_i P_{i-1}' P_{j', v'} = P_i P_{j'}$ implies $P_{i-1}' = P_{j'}$, a contradiction, there are $k - 1$ additional parallel edges between $v_{i-1}'$ and $v_i'$. Since the same idea works when $P_{i-1}'$ is replaced with any other of the $k$ labels, the number of parallel edges from $v_{i-1}'$ to $v_i'$ is greater than or equal to $k$. Running the argument backward shows that for each edge between $v_{i-1}'$ to $v_i'$, there is a corresponding edge from $\emptyset$ to $\emptyset$. 


Since the number of these is $k$, we have that the number of parallel edges from $v'_i$ to $v'_j$ is exactly $k$.

**Corollary 5:** For $0 \leq i \leq n$, $\dim S^1_{v'_i} + \dim S^1_{j} \geq 2k$.

**Proof:** Starting with $\dim S_{v'_i} = n + k$ and repeatedly applying Lemma 2 gives $\dim S^1_{v'_i} \geq n + k - 2i$ and, likewise, $\dim S^1_{j} \geq -n + k + 2i$. Combining these two equations gives the desired result.

**Lemma 3:** Any two minimal trellises for the same stabilizer code are isomorphic.

**Proof:** Since $\dim S^1_{v'_i}$ and $\dim S^1_{j}$ are invariant, we know from Corollary 2 that any trellises satisfying Theorem 1 have the same vertex degrees and are, hence, isomorphic. It remains to specify the mapping. Let there be two minimal trellises for the code with sets $V, E, V', E'$, respectively. Fix a $v \in V_i$ and consider its past coset (see Fig. 3). Pick a Pauli string in $S^1_{v'_i}$ in this coset and determine its terminus $v' \in V'_{v}$. The map $f : V \rightarrow V'$ and $f(v) = v'$ is clearly an isomorphism. It immediately follows that the edge isomorphism is given by $(s(e), P_i, t(e)) \mapsto (f(s(e)), P_i, f(t(e)))$.

**Proposition 3:** A set of Pauli strings is in TOF if and only if it has the left–right property.

**Proof:** Suppose that a set of Pauli strings is in TOF but does not have the left–right property. Then, there exists two elements of the set, $\hat{P}$ and $\bar{P}$, such that $L(\hat{P}) = L(\bar{P})$ or $R(\hat{P}) = R(\bar{P})$. Without loss of generality, assume that $L(\hat{P}) = L(\bar{P})$ and $R(\hat{P}) > R(\bar{P})$. One can always replace any power of $X$ or $Z$ with the first power by repeatedly applying it to itself since $p$ is prime and any power generates the cyclic group $\langle p \rangle$. Assume that this is done for $\hat{P}$ and replace the power of $X$ or $Z$ in $\hat{P}$ by $X^{p^i-1}$ or $Z^{p^i-1}$, $\hat{P} \mapsto P$, and $\bar{P} \mapsto P'$. Then, $PP'$ has lower span length than $\hat{P}$, a contradiction to the fact that the set is in TOF.

Now, suppose that the set has the left–right property but is not in TOF. The only way to reduce the span length of the set is to increase a left index or decrease a right index. However, this is impossible since there is only a single power of $X$ or $Z$ at these indices by the left–right property. Hence, the set already has the lowest total span length.

**Proposition 5:** The Viterbi algorithm solves (10) for independently and identically distributed noise models.

**Proof:** Let $\mu(v)$ be the value of the optimal (minimum-weight) path at vertex $v$, initialize $\mu(0) = 0$ at $V_0$, and denote by $wt(e)$ the weight of edge $e$ (see Algorithm 1). We proceed by induction on the depth $i$. By construction, all paths from $V_0$ to $v$ in $V_1$ consist of a single edge. The optimal path $P_0(v)$ to $v$ is given by the minimum-weighted edge from $V_0$ to $v$. By Algorithm 1, $\mu(v) = \min_{e \in E_i} \{\mu(s(e)) + wt(e)\} = wt(e)$ for the edge $e \in E_i$ of minimum weight, which is correct. Now, assume that the Viterbi algorithm correctly computes the optimal path for all depths $1$ to $i$ and let $v \in V_{i+1}$.

Let $\mu(v) = \min_{e \in E_i} \{\mu(s(e)) + wt(e)\}$

$$\mu(v) = \min_{e \in E_i} \{\mu(s(e)) + wt(e)\}$$

Every path from $V_0$ to $v$ is of this form.

**Theorem 3** ([56, Th. 2.10]): The Viterbi algorithm requires $O(|E|)$ arithmetic operations.

**Proof:** It suffices to characterize the operations of line 14 in Algorithm 1. For each vertex $v \in V_i$ for $1 \leq i \leq n$, $deg_{in}(v)$ additions are preformed

$$\text{Total additions} = \sum_{i=1}^{n} \sum_{v \in V_i} \text{deg}_{in}(v).$$

For simplicity, we take a computational model, where $x - 1$ minimums are computed for a list of size $x$. From this, we then have

$$\text{Total minimums} = \sum_{i=1}^{n} \sum_{v \in V_i} \text{deg}_{out}(v) - \sum_{i=1}^{n} \sum_{v \in V_i} 1.$$

The first term in this sum is equal to the total number of edges, $|E|$, and the second term is the total number of vertices minus $V_0$, $|V| - 1$.

$$\text{Total additions} = |E| - |V| + 1.$$
\[ = 1 + \sum_{j=0}^{i-1} \sum_{v \in V_j} P_j(v) (\deg_{\text{out}}(v) - 1). \]

**Proposition 6:** Let unprimed quantities be with respect to the syndrome trellis and primed quantities be with respect to any other trellis for the same code. Denote by

\[ \mathcal{E}_j = \sum_{v \in V_j} (\deg_{\text{out}}(v) - 1) \]

the number of expansions at depth \( j \) and likewise for the primes, and define

\[ \Delta_i := \sum_{j=0}^{i-1} p \dim S_{\mathcal{F}_j}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

Then, for \( R_\kappa < i \leq R_{\kappa+1} \),

\[ \sum_{j=0}^{i-1} p \dim S_{\mathcal{F}_{j-1}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) = \Delta_i + \sum_{a=1}^{\kappa} p_{\kappa,a}(p_a - 1)\Delta R_a \]

where \( p_a = p \dim S_{\mathcal{F}_0}^\perp - \dim S_{\mathcal{F}_{R_1}}^\perp \), and \( p_{\kappa,a} = p \dim S_{\mathcal{F}_{R_k}}^\perp - \dim S_{\mathcal{F}_{R_k+1}}^\perp \).

**Proof:** We proceed by induction on \( \kappa \). For \( \kappa = 1 \)

\[ R_1 < i \leq R_2 \]

\[ \dim S_{\mathcal{F}_{R_0}}^\perp = 0 \]

\[ \dim S_{\mathcal{F}_{R_1}}^\perp = \dim S_{\mathcal{F}_{R_1}}^\perp \]

\[ p_{1,1} = p \dim S_{\mathcal{F}_{R_1}}^\perp - \dim S_{\mathcal{F}_{R_0}}^\perp = 1 \]

\[ p_1 = p \dim S_{\mathcal{F}_{R_1}}^\perp - \dim S_{\mathcal{F}_{R_0}}^\perp = p \dim S_{\mathcal{F}_{R_1}}^\perp. \]

Then

\[ \sum_{j=0}^{i-1} p \dim S_{\mathcal{F}_{j-1}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ = \sum_{j=0}^{R_{\kappa}-1} p \dim S_{\mathcal{F}_{j-1}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) + \sum_{j=R_{\kappa}}^{i-1} p \dim S_{\mathcal{F}_{j-1}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ = p \dim S_{\mathcal{F}_{R_1}}^\perp \sum_{j=0}^{R_{\kappa}-1} (\mathcal{E}_j' - \mathcal{E}_j) + \sum_{j=R_{\kappa}}^{i-1} p \dim S_{\mathcal{F}_{j}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ = p \dim S_{\mathcal{F}_{R_1}}^\perp - \dim S_{\mathcal{F}_{R_0}}^\perp \sum_{j=0}^{R_{\kappa}-1} p \dim S_{\mathcal{F}_{j}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ - \sum_{j=0}^{R_{\kappa}-1} \dim S_{\mathcal{F}_j}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ + \sum_{j=0}^{R_{\kappa}-1} \dim S_{\mathcal{F}_j}^\perp (\mathcal{E}_j' - \mathcal{E}_j) + \sum_{j=R_{\kappa}}^{i-1} \dim S_{\mathcal{F}_j}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ = p_1 \sum_{j=0}^{R_{\kappa}-1} \dim S_{\mathcal{F}_j}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ + \sum_{j=0}^{R_{\kappa}-1} \dim S_{\mathcal{F}_j}^\perp (\mathcal{E}_j' - \mathcal{E}_j) = (p_1 - 1) \sum_{j=0}^{R_{\kappa}-1} \dim S_{\mathcal{F}_j}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ + \Delta_i = p_{1,1}(p_a - 1)\Delta R_a \]

Assume that the result is true for \( \kappa - 1 \). We wish to show that

\[ \sum_{j=0}^{i-1} p \dim S_{\mathcal{F}_{j-1}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) = \Delta_i + \sum_{a=1}^{\kappa} p_{\kappa,a}(p_a - 1)\Delta R_a \]

where

\[ R_\kappa < i \leq R_{\kappa+1} \]

\[ p_{\kappa,a} = p \dim S_{\mathcal{F}_{R_k}}^\perp - \dim S_{\mathcal{F}_{R_k+1}}^\perp \]

\[ p_a = p \dim S_{\mathcal{F}_{R_k}}^\perp - \dim S_{\mathcal{F}_{R_k+1}}^\perp. \]

Then, we have

\[ \sum_{j=0}^{i-1} p \dim S_{\mathcal{F}_{j-1}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ = \sum_{j=0}^{R_{\kappa}-1} p \dim S_{\mathcal{F}_{j-1}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) + \sum_{j=R_{\kappa}}^{i-1} p \dim S_{\mathcal{F}_{j-1}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ = \sum_{j=0}^{R_{\kappa}-1} p \dim S_{\mathcal{F}_{j}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) + \sum_{j=R_{\kappa}}^{i-1} p \dim S_{\mathcal{F}_{j}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ = \sum_{j=0}^{R_{\kappa}-1} p \dim S_{\mathcal{F}_{j}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ + \sum_{j=R_{\kappa}}^{i-1} p \dim S_{\mathcal{F}_{j}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ = \sum_{j=R_{\kappa}}^{i-1} p \dim S_{\mathcal{F}_{j}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ = \sum_{a=1}^{\kappa} p_{\kappa,a}(p_a - 1)\Delta R_a \]

\[ \sum_{j=0}^{i-1} p \dim S_{\mathcal{F}_{j-1}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ = \sum_{j=0}^{i-1} p \dim S_{\mathcal{F}_{j}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ + \sum_{j=R_{\kappa}}^{i-1} p \dim S_{\mathcal{F}_{j}}^\perp (\mathcal{E}_j' - \mathcal{E}_j) \]

\[ = \sum_{a=1}^{\kappa} p_{\kappa,a}(p_a - 1)\Delta R_a \]
\[ p_x \Delta R_k - \Delta R_k + \sum_{a=1}^{k-1} p^\text{dim} S_{R_k}^{j}(p_a - 1) \Delta R_a + \sum_{j=R_k}^{i-1} p^\text{dim} S_{j}^{j}(\mathcal{E}_j' - \mathcal{E}_j) \]

where the last line follows because every term in the sum is positive.

**Theorem 5:** For \( S = (s_1, \ldots, s_{n-k}) \), the minimal trellis for \( S \) is given by the trellis product of the minimal trellises for each \( s_i \).

**Proof:** We show minimality, which is the nontrivial part of the proof.

We proceed by induction on the number of generators in the product. Let \( \tilde{S}_i \) be the subset of generators of \( S^\perp \) (in TOF), which have trivial partial syndromes with respect to \( s_i \). Let \([L_1, R_1]\) and \([L_2, R_2]\) be the spans of \( s_1 \) and \( s_2 \), respectively, and construct trellises for each following Section II with vertex profiles:

\[ |V_i^1| = \begin{cases} 1, & \text{if } i \in [0, L_1) \\ p^\text{dim} S^\perp - \text{dim} S_i^\perp, & \text{if } i \in [L_1, R_1) \\ 1, & \text{if } i \in [R_1, n] \end{cases} \]

and

\[ |V_i^2| = \begin{cases} 1, & \text{if } i \in [0, L_2) \\ p^\text{dim} S^\perp - \text{dim} S_i^\perp, & \text{if } i \in [L_2, R_2) \\ 1, & \text{if } i \in [R_2, n] \end{cases} \]

The vertex profiles of the trellis product of the trellises for \( s_1 \) and \( s_2 \) are given by Lemma 5(i) to be as shown at the top of the next page, where we have used that for sets \( A, B \subset C, (C \setminus A) \cup (C \setminus B) = C \setminus (A \cap B) \), \( \tilde{S}_{i_1 \cap \cdots \cap i_{n-k-1}} := \tilde{S}_{i_1} \cap \cdots \cap \tilde{S}_{i_{n-k-1}} \), and \( \text{the past and future are taken with respect to the generators } s_1 \) and \( s_2 \), and the intersection \( \tilde{S}_{i_1} \cap \cdots \cap \tilde{S}_{i_{n-k-1}} \) depends on \( i \). Having recovered the minimal form of Theorem 4, assume that the theorem holds for the trellis product of \( s_1, \ldots, s_{n-k-1} \). It remains to show that the overlap \(|V_i^{1,\ldots,n-k-1}/|V_n^{n-k-1}| \) produces (4). In this case, \( \tilde{S}_{1 \cap \cdots \cap i_{n-k-1}} \neq \emptyset \) so

\[ |V_i^{1,\ldots,n-k-1}/|V_n^{n-k-1}| = |V_i^{1,\ldots,n-k-1}/|p^\text{dim} S^\perp - \text{dim} S_i^\perp| \]

The proof for the edges is similar.

**Proposition 7:** Let \(|E|\) be the total number of edges in a trellis diagram for a CSS code whose \( X \) and \( Z \) codes have trellises with a total number of edges \(|E_X|\) and \(|E_Z|\), respectively. Then

\[ |E_X| + |E_Z| \leq |E| \leq |E_X||E_Z| - p^2n(n-1). \] (19)

**Proof:** Let \( \xi_i = \log p \sum e \) and \( \eta_i = \log p \sum e \) such that \( |E_X| = \sum_{i=1}^{n} \xi_i \) and \( |E_Z| = \sum_{i=1}^{n} \eta_i \), and recall that \( \xi_i, \eta_i \geq 1 \). By Lemma 5, \( |E| = \sum_{i=1}^{n} \xi_i + \eta_i \). The left inequality is clear. For the right, we have

\[ (|E_X| + |E_Z|)^2 = |E_X|^2 + |E_Z|^2 + 2|E_X||E_Z| = |E_X|^2 + |E_Z|^2 \]
The author Eric Sabo would like to thank Evans Harrell and Benjamin Ide for helpful discussions.

REFERENCES

[1] E. Berlekamp, R. McEliece, and H. Van Tilborg, “On the inherent intractability of certain coding problems (corresp.),” *IEEE Trans. Inf. Theory*, vol. IT-24, no. 3, pp. 384–386, May 1978, doi: 10.1109/TIT.1978.1055873.

[2] P. Iyer and D. Poulin, “Hardness of decoding quantum stabilizer codes,” *IEEE Trans. Inf. Theory*, vol. 61, no. 9, pp. 5209–5223, Sep. 2015, doi: 10.1109/TIT.2015.2422294.

[3] A. G. Fowler, “Minimum weight perfect matching of fault-tolerant topological quantum error correction in average O(1) parallel time,” 2013, arXiv:1307.1740, doi: 10.48550/arXiv.1307.1740.

[4] A. M. Stephens, “Fault-tolerant thresholds for quantum error correction with the surface code,” *Phys. Rev. A*, vol. 89, no. 2, 2014, Art. no. 022321, doi: 10.1103/PhysRevA.89.022321.

[5] J. Wootton, “A simple decoder for topological codes,” *Entropy*, vol. 17, no. 4, pp. 1946–1957, 2015, doi: 10.3390/e17041946.

[6] D. S. Wang, A. G. Fowler, C. D. Hill, and L. C. L. Hollenberg, “Graphical algorithms and threshold error rates for the 2D colour code,” 2009, arXiv:0907.1708, doi: 10.48550/arXiv.0907.1708.

[7] H. Bombin, G. Duclos-Cianci, and D. Poulin, “Universal topological phase of two-dimensional stabilizer codes,” *New J. Phys.*, vol. 14, no. 7, 2012, Art. no. 073048, doi: 10.1088/1367-2630/14/7/073048.

[8] N. Delfosse, “Decoding color codes by projection onto surface codes,” *Phys. Rev. A*, vol. 89, no. 1, 2014, Art. no. 012317, doi: 10.1103/PhysRevA.89.012317.

[9] A. M. Stephens, “Efficient fault-tolerant decoding of topological color codes,” 2014, arXiv:1402.3037, doi: 10.48550/arXiv.1402.3037.

[10] A. B. Alishouse, A. N. Bhagoji, and P. K. Sarvepalli, “On the local equivalence of 2D color codes and surface codes with applications,” 2018, arXiv:1804.00866, doi: 10.48550/arXiv.1804.00866.

[11] A. Kubica and N. Delfosse, “Efficient color code decoders in $d \geq 2$ dimensions from toric code decoders,” 2019, arXiv:1905.07393, doi: 10.48550/arXiv.1905.07393.

[12] B. J. Brown, “Conservation laws and quantum error correction: Towards a generalised matching decoder,” *IEEE Bits Inf. Theory Mag.*, vol. 2, no. 3, pp. 5–19, Dec. 2022, doi: 10.1109/MBIT.2023.3246025.

[13] N. Delfosse and N. H. Nickerson, “Almost-linear time decoding algorithm for topological codes,” 2017, arXiv:1709.06218, doi: 10.48550/arXiv.1709.06218.

[14] Q. Huang, M. Newman, and K. R. Brown, “Fault-tolerant weighted union-find decoding on the toric code,” *Phys. Rev. A*, vol. 102, no. 1, 2020, Art. no. 012419, doi: 10.1103/PhysRevA.102.012419.

[15] N. Delfosse and M. B. Hastings, “Union-find decoders for holographic product codes,” *Quantum*, vol. 5, 2021, Art. no. 406, doi: 10.22331/q-2021-03-10-406.

[16] N. Delfosse, V. Londe, and M. Beverland, “Towards a union-find decoder for quantum LDPC codes,” *IEEE Trans. Inf. Theory*, vol. 68, no. 5, pp. 3187–3199, May 2021, doi: 10.1109/TIT.2022.3143452.

[17] M. Herold, E. T. Campbell, J. Eisert, and M. J. Kastoryano, “Cellular-automaton decoders for topological quantum memories,” *npj Quantum Inf.*, vol. 1, no. 1, pp. 1–8, 2015, doi: 10.1038/npjqi.2015.10.

[18] M. Herold, M. J. Kastoryano, E. T. Campbell, and J. Eisert, “Cellular automaton decoders of topological quantum memories in the fault tolerant setting,” *New J. Phys.*, vol. 19, no. 6, 2017, Art. no. 063012, doi: 10.1088/1367-2630/aa7099.

[19] A. Kubica and J. Preskill, “Cellular-automaton decoders with provable thresholds for topological codes,” *Phys. Rev. Lett.*, vol. 123, no. 2, 2019, Art. no. 020501, doi: 10.1103/PhysRevLett.123.020501.

[20] A. J. Landahl, J. T. Anderson, and P. R. Rice, “Fault-tolerant quantum computing with color codes,” 2011, arXiv:1108.5738, doi: 10.48550/arXiv.1108.5738.

[21] G. Duclos-Cianci and D. Poulin, “Fast decoders for topological quantum codes,” *Phys. Rev. Lett.*, vol. 104, no. 5, 2010, Art. no. 050504, doi: 10.1103/PhysRevLett.104.050504.

[22] G. Duclos-Cianci and D. Poulin, “A renormalization group decoding algorithm for topological quantum codes,” in *Proc. IEEE Inf. Theory Workshop*, 2010, pp. 1–5, doi: 10.1109/CIG.2010.5952866.

[23] S. Bravyi and D. Poulin, “Efficient decoding of topological quantum codes,” *Phys. Rev. A*, vol. 85, no. 2, 2012, Art. no. 022317, doi: 10.1103/PhysRevA.85.022317.

[24] L. Berent, L. Burgolhar, P.-H. S. Derks, J. Eisert, and R. Wille, “Decoding quantum color codes with MAXSAT,” Jul. 2023, arXiv:2303.14237, doi: 10.48550/arXiv.2303.14237.

[25] D. J. MacKay, G. Mitchison, and P. L. McFadden, “Sparse-graph codes for quantum error correction,” *IEEE Trans. Inf. Theory*, vol. 50, no. 10, pp. 2315–2330, Oct. 2004, doi: 10.1109/TIT.2004.834737.

[26] D. Poulin and Y. Chung, “On the iterative decoding of sparse quantum codes,” *Quantum Inf. Comput.*, vol. 8, pp. 987–1000, 2008, doi: 10.5555/2016985.2016993.

[27] S. Bravyi, M. Suchara, and A. Vargo, “Efficient algorithms for maximum likelihood decoding in the surface code,” *Phys. Rev. A*, vol. 90, no. 3, 2014, Art. no. 032326, doi: 10.1103/PhysRevA.90.032326.

[28] A. J. Ferris and D. Poulin, “Tensor networks and quantum error correction,” *Phys. Rev. Lett.*, vol. 113, no. 3, 2014, Art. no. 030501, doi: 10.1103/PhysRevLett.113.030501.

[29] A. S. Darmawan and D. Poulin, “Tensor-network simulations of the surface code under realistic noise,” *Phys. Rev. Lett.*, vol. 119, no. 4, 2017, Art. no. 040502, doi: 10.1103/PhysRevLett.119.040502.

[30] A. S. Darmawan and D. Poulin, “Linear-time general decoding algorithm for the surface code,” *Phys. Rev. E*, vol. 97, no. 5, 2018, Art. no. 051302, doi: 10.1103/PhysRevE.97.051302.
Eric Sabo received the B.S. and M.S. degrees in applied mathematics and high-energy physics from the University of Delaware, Newark, DE, USA in 2014, and the Ph.D. degree in mathematics from the Georgia Institute of Technology, Atlanta, GA, USA, in 2022. His research interests include classical and quantum coding theory and quantum computation.

Arun B. Aloshious (Member, IEEE) received the B.E. degree in electronics and communication engineering from Anna University, Chennai, India, in 2009, the M.Tech. degree in electronics and communication engineering from the National Institute of Technology, Calicut, India, in 2011, and the Ph.D. degree in electrical engineering from Indian Institute of Technology Madras, Chennai, in 2020.

Between 2020 and 2021, he was a Postdoctoral Research Associate with Duke University, Durham, NC, USA. He is currently an Assistant Professor with the Indian Institute of Technology Guwahati, Guwahati, India. His research interests include classical and quantum coding theory, quantum computation, information theory, signal processing, and communication.

Kenneth R. Brown (Senior Member, IEEE) received the B.S. degree in chemistry from the University of Puget Sound, Tacoma, WA, USA, in 1998, and the Ph.D. degree in chemistry from the University of California-Berkeley, Berkeley, CA, USA, in 2003.

He is currently the Michael J. Fitzpatrick Distinguished Professor of electrical and computer engineering with Duke University, Durham, NC, USA. His research interests include quantum control, quantum error correction, and ion trap quantum devices.

[77] A. G. Fowler, A. C. Whiteside, and L. C. Hollenberg, “Towards practical classical processing for the surface code,” Phys. Rev. Lett., vol. 108, no. 18, 2012, Art. no. 180501, doi: 10.1103/PhysRevLett.108.180501.

[78] B. Rahn, A. C. Doherty, and H. Mabuchi, “Exact performance of concatenated quantum codes,” Phys. Rev. A, vol. 66, no. 3, Sep. 2002, Art. no. 032304, doi: 10.1103/PhysRevA.66.032304.

[79] D. Poulin, “Optimal and efficient decoding of concatenated quantum block codes,” Phys. Rev. A, vol. 74, Nov. 2006, Art. no. 052333, doi: 10.1103/PhysRevA.74.052333.

[80] V. V. Zyablov and V. R. Sidorenko, “Bounds on complexity of trellis decoding of linear block codes (in Russian),” Probl. Peredachi Inf., vol. 29, no. 3, pp. 3–9, 1993. [Online]. Available: https://www.mathnet.ru/php/archive.phtml?wshow=paper&jrnid=ppi&paperid=183&option_lang=eng

[81] T. Kasami, T. Takata, T. Fujiwara, and S. Lin, “On complexity of trellis structure of linear block codes,” IEEE Trans. Inf. Theory, vol. 39, no. 3, pp. 1057–1064, May 1993, doi: 10.1109/18.256515.

[82] A. Lafourcade and A. Vardy, “Lower bounds on trellis complexity of block codes,” IEEE Trans. Inf. Theory, vol. 41, no. 6, pp. 1938–1954, Nov. 1995, doi: 10.1109/18.476318.

[83] A. Lafourcade and A. Vardy, “Asymptotically good codes have infinite trellis complexity,” IEEE Trans. Inf. Theory, vol. 41, no. 2, pp. 555–559, Mar. 1995, doi: 10.1109/18.370171.

[84] Ø. Ytrehus, “On the trellis complexity of certain binary linear block codes,” IEEE Trans. Inf. Theory, vol. 41, no. 2, pp. 559–560, Mar. 1995, doi: 10.1109/18.370172.

[85] E. Bertrand and F. Labeau, “Simplified trellis decoding of block codes by selective pruning,” in Proc. Conf. Rec. 38th Asilomar Conf. Signals, Syst. Comput., 2004, pp. 1115–1119, doi: 10.1109/ACSSC.2004.1399314.

[86] T. Fujiwara, H. Yamamoto, T. Kasami, and S. Lin, “A trellis-based recursive maximum-likelihood decoding algorithm for binary linear block codes,” IEEE Trans. Inf. Theory, vol. 44, no. 2, pp. 714–729, Mar. 1998, doi: 10.1109/18.661515.

[87] R. Morelos-Zaragoza, T. Fujiwara, T. Kasami, and S. Lin, “Constructions of generalized concatenated codes and their trellis-based decoding complexity,” IEEE Trans. Inf. Theory, vol. 45, no. 2, pp. 725–731, Mar. 1999, doi: 10.1109/18.749022.