An Efficient Algorithm for Constructing Minimal Trellises for Codes over Finite Abelian Groups

Vijay V. Vazirani, Huzur Saran, and B. Sundar Rajan, Member, IEEE

Abstract—We present an efficient algorithm for computing the minimal trellis for a group code over a finite abelian group, given a generator matrix for the code. We also show how to compute a succinct representation of the minimal trellis for such a code, and present algorithms that use this information to compute efficiently local descriptions of the minimal trellis. This extends the work of Kschischang and Sorokine, who treated the case of linear codes over fields. An important application of our algorithms is to the construction of minimal trellises for lattices.

A key step in our work is handling codes over cyclic groups $C_p^n$, where $p$ is a prime. Such a code can be viewed as a module over the ring $Z_p^n$. Because of the presence of zero divisors in the ring, modules do not share the useful properties of vector spaces. We get around this difficulty by restricting the notion of linear combination to a $p$-linear combination, and by introducing the notion of a $p$-generator sequence, which enjoys properties similar to those of a generator matrix for a vector space.

Index Terms—Trellises, group codes, codes over rings, lattices, algorithms, Gaussian elimination.

I. INTRODUCTION

Following the success of trellis-coded modulation [23] (which revolutionized the transmission rates of modems in bandwidth-limited channels), researchers have been studying block-coded modulation [7], [13], and [15]. Block group codes are basic ingredients in a large class of block-coded modulation schemes [7]. The coding gain promised by these schemes can be achieved only with soft-decision decoding, [22] and [23]. Trellises provide a general framework for efficient soft-decision decoding of codes [26], for instance by using the Viterbi algorithm [4]. Since the decoding effort is directly related to the size of the trellis, much work has been devoted to characterizing and constructing minimal trellises for group codes [6], [9], [16], [17], and [21].

In this paper, we present an $O(k^2n + s)$ time algorithm for constructing the minimal trellis of a block code over a finite abelian group, given a generator matrix for the code, where $n$ is the length of the code, $k$ is the number of rows in the generator matrix, and $s$ is the number of states in the minimal trellis. (Throughout the paper, we will assume that it takes one unit of time to perform an operation over the underlying field, ring, or group.) For decoding purposes, it is perhaps more important to be able to compute efficiently local descriptions of the minimal trellis. For this purpose, we show how a succinct description of the minimal trellis for such codes can be computed in $O(k^2n)$ time and $O(kn)$ space; notice that this is polynomial in $n$ time and space, even though the minimal trellis may be exponentially large. We give algorithms that use this information to compute, for example, all transitions into or out of a state in $O(k)$ time.

Perhaps the most important application of our work is to the construction of minimal trellises for lattices, since this problem essentially reduces to that of constructing minimal trellises for block codes over abelian groups, [2] and [6]. This is elaborated in Section XI. Another application arises as follows. Certain famous nonlinear binary codes (including Kerdock, Preparata, and Goethals codes) contain more codewords than any known linear code of the same length. In a recent breakthrough, Hammons, Kumar, Calderbank, Sloane, and Solé have shown that under the Gray map from $(Z_2)^2$ to the ring $Z_4$, these codes are linear over $Z_4$ [11], or equivalently, are group codes over $C_4$.

We have built directly on the work of Forney and Trott [9] and Kschischang and Sorokine [16]. Forney and Trott, using the work of Willems on dynamical systems [24], [25], show that group codes admit unique minimal trellises. Furthermore, they present important structural properties of such trellises, especially in their State Space Theorem (see Section III). Kschischang and Sorokine have given an $O(k^2n + s)$ time algorithm for constructing the minimal trellis for a linear code over a field, given a generator matrix for the code (see Section IV). They also present an efficient algorithm for computing local descriptions of the minimal trellis.

The essential step in the algorithm of Kschischang and Sorokine is obtaining a special generator matrix for the code: a two-way proper generator matrix. We give a simple proof that such a generator matrix yields a minimal trellis (Section IV). A key step toward extending this result to codes over finite abelian groups is handling codes over cyclic groups $C_p^n$, where $p$ is a prime. Such codes can be viewed as linear codes over the ring $Z_p^n$ and are therefore modules over $Z_p^n$. The extension is not straightforward; the main problem is the presence of zero divisors in the ring. In Section V, we outline the difficulties encountered because of zero divisors. Some of these are quite general, e.g., the difficulty in giving satisfactory definitions for basis and dimension of modules over $Z_p^n$. Others are specific to minimal trellises.
We then introduce the notions of \( p \)-linear combinations and \( p \)-generator sequences that enable us to get around these difficulties (Section VI). We show how Gaussian elimination can be adapted to this setting, and can be used for obtaining certain operations on modules over \( \mathbb{Z}_{pa} \), given an ordinary generator matrix for it. These notions should find other applications as well, since they enable one to perform certain operations on modules over \( \mathbb{Z}_{pa} \) similar to the way in which these operations are performed on vector spaces.

In Section VII, we give a natural generalization of a two-way proper matrix: namely, a two-way proper \( p \)-generator sequence, and we show how Gaussian elimination can be used to obtain it. Once this is done, a minimal trellis for a linear code, and from this trellis, using sectionalization, we obtain a minimal trellis for a group code over \( \mathbb{Z}_{pa} \) can be constructed in essentially the same manner as in the field case.

Finally, in Section VIII, we consider codes over finite abelian groups. First, we show that group codes over elementary abelian groups can be seen as linear codes over an appropriate finite field. We obtain a minimal trellis for this linear code, and from this trellis, using sectionalization, we obtain a minimal trellis for the given group code. To deal with arbitrary finite abelian groups, we show that it is sufficient to consider abelian \( p \)-groups. A code over such a group is in turn a linear code over a ring \( \mathbb{Z}_{pa} \), and can be handled analogously.

The problem of computing local descriptions of minimal trellises is addressed in Section IX. Two types of problems are solved. First, given two states at successive times, determine whether there is a transition between them, and if so, determine the set of labels on the transition. Second, given a state at time \( i \), compute all transitions into it and out of it.

In Section X, we build on the State Space Theorem to give algebraic structural properties of the set of transitions between two times in the minimal trellis for a group code; we call this the Transition Space Theorem. This theorem also defines a succinct representation for the minimal trellis of a group code, from which local descriptions can be computed. This applies to group codes over nonabelian groups as well; however, in general, the size of the representation may be super-polynomial.

II. PRELIMINARIES

In this paper, we will deal only with block codes, i.e., codes for which each codeword has the same length, \( n \). Let \( G \) be a finite abelian group, and let \( W = G^n \) be the \( n \)-fold direct product of \( G \). A subgroup \( C \) of \( W \) under the componentwise addition operation of \( G \) is said to be a group code over \( G \).

Let \( I \) denote the set of positive integers from 1 to \( n \); \( I \) will be called the time axis. An element \( a \in W \) will be called a sequence; \( a = (a_i, i \in I) \).

Let \( R \) be a ring; as a special case, \( R \) may also be a field. As before, let \( W = R^n \). Let \( C \) be a subgroup of \( W \) under the componentwise addition operation of \( R \), and assume furthermore that \( C \) is closed under componentwise multiplication with elements of \( R \). Then, \( C \) is said to be a linear code over \( R \). Clearly, the class of linear codes over fields is contained in the class of linear codes over rings, which is in turn contained in the class of group codes.

A trellis \( T \) for a group code \( C \) is an edge-labeled directed layered graph. The vertices of \( T \) are partitioned into disjoint subsets \( V_0, V_1, \ldots, V_n \). The set \( V_i \) is referred to as the set of states at time \( i \). \( V_0 \) contains a unique starting state \( v_0 \), and \( V_n \) contains a unique terminating state \( v_n \). Edges of \( T \) are allowed to connect only states with successive time indices. A transition \((u \to v)\), \( u \in V_i, v \in V_{i+1} \) is labeled with a nonempty subset of elements from the group \( G \). This transition is said to be out of state \( u \) and into state \( v \). A state having more than one out-transition (in-transition) will be called a forking state (collapsing state). A state \( u \) in a trellis \( T \) will be said to be forward proper (backward proper) if the sets of labels on the out-transitions (in-transitions) of \( u \) are pairwise disjoint.

Finally, a trellis \( T \) will be said to be two-way proper if each of its states is forward proper and backward proper.

A path from \( v_0 \) to \( v_n \) consists of \( n \) transitions,

\[
\text{where } v_i \in V_i. \text{ Such a path defines a set of words } (\alpha_1, \alpha_2, \ldots, \alpha_n), \text{ where each } \alpha_i \text{ is drawn from the set labeling the transition } (v_{i-1} \to v_i). \text{ We require that each state must be used, i.e., it must occur on some path from } v_0 \text{ to } v_n. \text{ Finally, we require that the set of all words defined by all paths in } T \text{ from } v_0 \text{ to } v_n \text{ be exactly the set of codewords in } C. \text{ We will say that state } s \text{ is responsible for all the codewords whose paths use state } s.

Clearly, there exists a trivial trellis for each group code \( C \); create a unique path from \( v_0 \) to \( v_n \) for each codeword, with unique intermediate states. Such a trellis will have as many states at each time as the number of codewords in \( C \). For several reasons, including efficient decoding, it is important to obtain a trellis for \( C \) having as few states as possible. We say that \( T \) is a minimal trellis for \( C \) if at each time, \( T \) has the smallest possible number of states.

Let \( C_1 \) and \( C_2 \) be two group codes over the same underlying group \( G \), and let \( T_1 \) and \( T_2 \) be trellises for these codes. Let \( C = C_1 \oplus C_2 \) be the direct sum of these two group codes. Since \( G \) is abelian, \( C \) is a group code. We define the operation of taking the product of trellises \( T_1 \) and \( T_2 \) to obtain a trellis \( T \) for the code \( C \) as in [16] as follows: Let \( \{U_i, 0 \leq i \leq n \text{ and } V_i, 0 \leq i \leq n \} \) be the sets of states of \( T_1 \) and \( T_2 \). The trellis \( T \) has states \( W_{ij} \), \( 0 \leq i \leq n \), where \( |W_i| = |U_i||V_i| \), and corresponding to each pair of states \( u \in U_i \) and \( v \in V_i \), there is a state \((u, v) \in W_i \). There is a transition from \((u, v) \in W_i \) to \((u', v') \in W_{i+1} \) if and only if (iff) \((u \to u') \) and \((v \to v') \) are transitions in \( T_1 \) and \( T_2 \), respectively. Let \( \alpha \) and \( \beta \) be the labels on the transitions \((u \to u') \) and \((v \to v') \). Then, the set of labels on transition \((u, v) \to (u', v') \) is \( \{a + b | a \in \alpha, b \in \beta\} \).

III. STRUCTURAL PROPERTIES OF MINIMAL TRELLISES FOR GROUP CODES

As established by Forney and Trott, structural properties of group codes lead to structural properties of minimal trellises for such codes. In this section, we will review properties essential for our work, especially those following from the State Space Theorem.
Let \( J \subseteq I \) be a subset of the minimal time axes. The projection map \( P_J: W \to W \) sends a sequence \( a \in W \) to the sequence \( b \) defined by

\[
b_i = \begin{cases} a_i, & \text{if } i \in J \\ 0, & \text{if } i \in I - J. \end{cases}
\]

Thus the projection map \( P_J \) simply “zeros out” the components of a sequence with indices in \( I - J \). Define the projection \( P_J(C) = \{ P_J(c) | c \in C \} \), as the image of \( C \) under the projection map \( P_J \). The projection map is a homomorphism, since \( P_J(a + b) = P_J(a) + P_J(b) \). Further, since \( C \) is a group, the image of \( C \) under \( P_J \), \( P_J(C) \), is a subgroup of \( W \). If \( J \) consists of the first \( k \) time indices, we will denote \( P_J(C) \) by \( P_{k-}(C) \), and \( P_{k-}(a) \) and \( P_{k-}(a) \) are similarly defined. \( P_{k-}(C) \) will be called the set of codeword pasts and \( P_{k+}(C) \) the set of codewords futures at time \( k \).

The cross section of \( C \) in \( J \), denoted by \( C_J \), is a subcode of \( C \) consisting of all codewords whose components in \( I - J \) are zero, i.e.,

\[
C_J = \{ c \in C | c_k = 0, \ k \in I - J \}.
\]

Notice that \( C_J \) is the kernel of the projection map \( P_{1-J} \) restricted to \( C \). Again, if \( J \) consists of the first \( k \) time indices, we will denote \( C_J \) by \( C_{k-} \) and \( C_{1-J} \) by \( C_{k+} \); these are called the past subcode and future subcode, respectively, in [9]. Since \( C_{k-} \) and \( C_{k+} \) are both normal subgroups of \( C \), \( C_{k-} + C_{k+} \) is also a normal subgroup of \( C \). Furthermore, since \( C_{k-} \cap C_{k+} = \{ 0 \} \), \( C_{k-} + C_{k+} \) is a direct sum.

The State Space Theorem of Forney and Trott [9] states that

\[
\frac{P_{k-}(C)}{C_{k-}} \cong \frac{P_{k+}(C)}{C_{k+}} \cong \frac{C}{C_{k-} + C_{k+}}.
\]

It will be instructive to consider the following bipartite graph \( H_k \), called the past-future graph at time \( k \). Its vertex sets are \( P_{k-}(C) \) and \( P_{k+}(C) \), and two vertices \( u \in P_{k-}(C) \) and \( v \in P_{k+}(C) \) are joined by an edge iff \( u + v \in C \). We will say that \((A, B), A \subseteq P_{k-}(C), B \subseteq P_{k+}(C) \) is a bipartite clique if for each \( u \in A \) and \( v \in B \), \((u, v) \) is an edge in \( H_k \). The State Space Theorem shows that \( H_k \) consists of disjoint bipartite cliques (see also [16]).

Notice that \( C_{k-} \subseteq P_{k-}(C) \) and \( C_{k+} \subseteq P_{k+}(C) \). Since \( C_{k-} \cap C_{k+} \) is a direct sum, there is a bipartite clique between the corresponding sets of vertices in \( H_k \). This clique will be called the zero clique since it corresponds to the subgroup \( C_{k-} \oplus C_{k+} \) of codewords of \( C \); one of its edges corresponds to the all-zero codeword.

For any \( c \in C \), consider the coset \( C_{k-} \oplus c_{k+} + c \). The codewords in this set consist of pasts corresponding to the elements of the coset \( P_{k-}(c) + P_{k-}(c) \) and futures corresponding to the elements of the coset \( P_{k+}(c) + P_{k+}(c) \). In \( H_k \), there is a bipartite clique between these sets of vertices; the edges of this clique correspond to \( C_{k-} \oplus C_{k+} + c \).

The construction of the unique minimal trellis \( T \) for \( C \) follows from the past–future graphs \( H_k, 1 \leq k < n \). \( T \) has \(|C|/(C_{k-} \oplus C_{k+})| \) states at time \( k \); each state is responsible for codewords in one of the cosets. The state that is responsible for codewords in the subgroup \( C_{k-} \oplus C_{k+} \) will be called the zero state, and will sometimes be denoted by 0. If \( u \) and \( v \) are states at times \( k \) and \( k + 1 \), respectively, then there is a transition from \( u \) to \( v \) iff the sets of codewords they are responsible for have a nonempty intersection, say \( A \). If so, the set of labels on this transition is the set of symbols in \( P_{(k+1)}(A) \), which we define as the projection of \( A \) onto the \((k + 1)\)th coordinate. Define the output group at time \( k + 1 \) to be \( G_{k+1} = P_{(k+1)}(C) \); obviously, \( G_{k+1} \) is a subgroup of \( G \).

IV. Minimal Trellises for Codes Over Fields

In this section we will describe the algorithm of Kschischang and Sorokine (KS) [16] for linear codes over fields, since we build directly on it. We will also give a simpler proof for the KS algorithm. The running time of the KS algorithm is \( O(k^2n + s) \), where the generator matrix has size \( k \times n \), and \( s \) is the number of states in the minimal trellis.

Our simplified proof of minimality relies on the following characterization established by Willems [24], [25] in the context of dynamical systems. We will use this characterization for proving the minimality of trellises in the cyclic group and abelian group cases as well.

Theorem 4.1 Willems [24] and [25]: A two-way proper trellis for a block code \( C \) is the unique minimal trellis for \( C \).

For a simplified proof of Theorem 4.1, proven in the context of minimal trellises for group codes, see [18] and [19]. For further extensions of Willems’ results to codes over finite abelian groups, see [3]. In general, a code may not admit a two-way proper trellis. However, if such a trellis does exist for the code, it is guaranteed to be minimal; see [14] for a proof.

A linear code \( C \) over a field \( \text{GF}(q) \) is a vector space, and can be described by a generator matrix \( A \). The minimal trellis for the code generated by a single row vector of \( A \) can be obtained in a straightforward manner as explained below. Since \( C \) is the sum of the codes generated by the rows of \( A \), the product of the trellises for these codes will be a trellis for \( C \); the operation of computing the product of trellises was introduced by Kschischang and Sorokine [16] for precisely this reason. In general, this trellis may not be minimal. Forney has called a generator matrix that gives rise to a minimal trellis a trellis-oriented generator matrix [6]. The key step in the algorithm of Kschischang and Sorokine is efficiently obtaining a trellis-oriented generator matrix, given an arbitrary generator matrix \( A \).

Let \((a_1 \ a_2 \ \cdots \ a_n)\) be a row of \( A \). Let \( a_i \) be the first nonzero entry and \( a_j \) the last nonzero entry in this row, i.e., \( a_k = 0 \) for \( k < i \) and for \( k > j \). Then, we will say that this row starts at \( i \) and ends at \( j \). Furthermore, \( a_i \) will be called the starting element of this row and \( a_j \) will be called its ending element. The minimal trellis for the code generated by this vector has a simple structure: It has a single forking state with \( q \) out-transitions at time \( i - 1 \), and a single collapsing state with \( q \) in-transitions at time \( j \); all other states have one in- and one out-transition; see Example 1.

Example 1: For the code generated by a single vector over \( \text{GF}(q) \), the minimal trellis consists of a forking state at the time at which this vector starts and a collapsing state at the time
at which this vector ends. For example, the minimal trellis for the code over $GF(5)$ generated by $(030210)$ is given in Fig. 1.

We will first prove that for establishing minimality of the product of two minimal trellises of two group codes, it is sufficient to establish two-way properness of the zero states at each time. The lemmas below consider only forward properness—analogous statements hold for backward properness. By the set of labels emanating from a state we mean the union of the sets of labels on all transitions out of this state.

**Lemma 4.2:** Let $T$ be a minimal trellis for a group code $C$ over a finite abelian group $G$, and let $s_0$ and $s$ be the zero state and an arbitrary state at time $i$. Let $a_0$ and $a$ be the sets of labels emanating from $s_0$ and $s$, respectively, and let $G_{i+1}$ be the output group at time $i + 1$. Then $a_0$ is a subgroup of $G_{i+1}$, and $a$ is an element of the quotient group $G_{i+1}/a_0$.

**Proof:** State $s_0$ is responsible for the set of codewords in $C_i^+ \oplus C_i$. Since $C_i^+ \oplus C_i$ is a subgroup of $C_i$, for $P(i+1)(C_i^+ \oplus C_i) \subseteq P(i+1)(C) = G_{i+1}$.

Since the set of codewords that $s$ is responsible for forms a coset of $C_i \oplus C_i$ in $C$, using a similar argument, we find that $a$ is an element of the quotient group $G_{i+1}/a_0$.

**Lemma 4.3:** Let $C_1$ and $C_2$ be group codes of length $n$ over the same underlying group $G$, and let $C = C_1 + C_2$. Let $T_1$ and $T_2$ be minimal trellises for $C_1$ and $C_2$, respectively, and let $T$ be the product of these trellises, which is a trellis for the code $C$. Let $a_0$ and $a_0 = \beta_0$ be the set of labels emanating from the zero states $s_1$ and $s_2$ at time $i$ in $T_1$ and $T_2$, respectively. Then, $T$ is forward proper at time $i$ if and only if $a_0$ and $a_0 = \beta_0$ intersect trivially, i.e., $a_0 \cap \beta_0 = \{0\}$, where $0$ is the identity element of $G$.

**Proof:** Since $a_0$ and $\beta_0$ are finite subgroups of $G$, by Lagrange's theorem

\[
[|a_0|][|\beta_0|] = |a_0 \cap \beta_0|.
\]

So the labels $(s_1, s_2)$ emanating from the zero state at time $i$ in $T$ are all distinct if $|a_0|\beta_0| = |a_0|\beta_0|$, which happens if and only if $a_0 \cap \beta_0 = \{0\}$.

Consider two arbitrary states $s_1$ and $s_2$ at time $i$ in $T_1$ and $T_2$, respectively. Let us view the set of labels emanating from $s_1$ as a coset of $a_0$, say $a + a_0$, and those emanating from $s_2$ as a coset of $\beta_0$, say $b + \beta_0$. Then, the label emanating from state $(s_1, s_2)$ in $T$ are given by $a_0 = \beta_0 + a + b$. As before, these will be all distinct if $|a_0\beta_0| = |a_0|\beta_0|$. The lemma follows.

**Remark:** Lemmas 4.2 and 4.3 extend in a straightforward way to group codes over an arbitrary finite group.

We say that a generator matrix is two-way proper if every row starts at a distinct point, and every row ends at a distinct point. The following is a restatement of Theorem 2 of Kschischang and Sorokine [16]; we give a simpler proof using Theorem 4.1 and Lemma 4.3.

**Theorem 4.4** Kschischang and Sorokine [16]: A generator matrix for a linear code over a field is trellis-oriented iff it is two-way proper.

**Proof:** Since multiplication by a nonzero element of a field is a one-to-one onto map, the set of symbols emanating from a zero state is either $\{0\}$ or the entire field. Compute the product of all the trellises of the rows of the generator matrix. At any time $i$, the zero state of the product trellis is forward proper iff the sets of labels emanating from zero states in the component trellises at time $i$ intersect trivially. This happens if and only if most one row of the generator matrix starts at $i$. A similar proof holds for backward properness.

Using two stages of Gaussian elimination, any generator matrix for $C$ can be converted into a two-way proper generator matrix. The first stage puts the matrix into the usual row echelon form, i.e., row $i + 1$ starts later than row $i$, for $1 \leq i < n - 1$. Then, by a process of “canceling upwards,” one can ensure that no two rows end at the same point; this process does not affect the starting points (see [16]).

**Example 2:** Consider the following generator matrix over $GF(2)$:

\[
\begin{pmatrix}
1100 \\
1010
\end{pmatrix}
\]

The trellises for the individual rows as well as the product trellis are shown in Fig. 2(a). In this case, the trellis obtained is not two-way proper. However, we may convert the above generator matrix to a two-way proper matrix to obtain

\[
\begin{pmatrix}
1100 \\
0110
\end{pmatrix}
\]

The minimal trellis obtained from this matrix is shown in Fig. 2(b).

**V. EXTENDING TO RINGS $\mathbb{Z}_{\text{p}^n}$:**

**The Difficulties Encountered**

A length-$n$ linear code $C$ over a ring $\mathbb{Z}_{\text{p}^n}$ is a submodule of the module $\mathbb{Z}_{\text{p}^n}$. Such a submodule can be specified via a
Let us point out some more general difficulties in working over modules, arising because of zero divisors (or more generally, the fact that the elements of the ring have different orders). There are two natural ways of defining linear dependence of a set of vectors $V$:

- a nontrivial linear combination of the vectors in $V$ gives the zero vector
- one of the vectors in $V$ can be expressed as a linear combination of the others.

In the case of a vector space, these two definitions are equivalent. However, in the case of a module, dependence in the first sense need not imply dependence in the second sense. For example, over $\mathbb{Z}_4$, the vectors $(12)$ and $(10)$ are dependent by the first definition, but not by the second.

Another problem is to define the dimension of a module. For example, over $\mathbb{Z}_4$, $(20)$ and $(02)$ form a basis for the module they generate. On the other hand, $(10)$ and $(01)$ form a basis for a module that strictly contains the first module. Consequently, defining the dimension of a module as the cardinality of its basis is not very meaningful.

VI. $p$-LINEAR COMBINATIONS AND $p$-GENERATOR SEQUENCES

In this section, we will introduce the notions of $p$-linear combinations and $p$-generator sequences, which enable us to avoid the difficulties mentioned in the previous section in working over $\mathbb{Z}_{p^n}$. We will show that $p$-linear combinations of $p$-generator sequences enjoy properties similar to those of a basis for a vector space: they uniquely generate the elements of a module, a suitable definition of the "dimension" of a module can be given, and the two notions of linear dependence become equivalent.

Let $V = \{\vec{v}_1, \cdots, \vec{v}_k\}$ be a set of vectors over $\mathbb{Z}_{p^n}$. We will say that

$$\sum_{i=1}^{k} a_i \vec{v}_i$$

is a $p$-linear combination of these vectors if all coefficients $a_i$ lie in $\mathbb{Z}_p = \{0, 1, \cdots, (p - 1)\}$. Notice that the elements $1, \cdots, (p - 1)$ are all units in $\mathbb{Z}_{p^n}$. We will denote by $p$-span$(V)$ the set of all vectors generated by $p$-linear combinations of vectors in $V$, and by span$(V)$ the set of vectors generated by ordinary linear combinations of vectors in $V$. We will say that a given linear combination (p-linear combination) uses vector $\vec{v}_i$ if the coefficient of $\vec{v}_i$ is nonzero in the linear combination (p-linear combination).

An ordered sequence of vectors $V = (\vec{v}_1, \cdots, \vec{v}_k)$ over $\mathbb{Z}_{p^n}$ is said to be a $p$-generator sequence if for $1 \leq i \leq k$, $p\vec{v}_i$ is a $p$-linear combination of the vectors $\vec{v}_{i+1}, \cdots, \vec{v}_k$ (in particular, $p\vec{v}_k$ is the zero vector). For each vector $\vec{v}_i$ one such $p$-linear combination is designated the canonical $p$-linear combination for $\vec{v}_i$. If $i < j$, we will say that $\vec{v}_i$ is earlier than $\vec{v}_j$ and that $\vec{v}_j$ is later than $\vec{v}_i$.

For an arbitrary set of vectors $V$, $p$-span$(V)$ may not be a module, for example if $p\vec{v}_i$ is not a $p$-linear combination of the vectors in $V$ for some $\vec{v}_i \in V$. However, ensuring this condition is not sufficient as shown in Example 4. On the other
hand, this condition together with the order among the vectors that is stated in the definition of a p-generator sequence turns out to be sufficient; this is established in Theorem 6.2.

**Example 4:** Consider the following set of vectors over \( Z_9 \):

\[
\begin{align*}
\vec{v}_1 &= (3106) \\
\vec{v}_2 &= (2270) \\
\vec{v}_3 &= (8510) \\
\vec{v}_4 &= (3533).
\end{align*}
\]

Here

\[
\begin{align*}
3\vec{v}_1 &= 2\vec{v}_2 + 2\vec{v}_3 \\
3\vec{v}_2 &= \vec{v}_1 + \vec{v}_4 \\
3\vec{v}_3 &= \vec{v}_1 + \vec{v}_4 \\
3\vec{v}_4 &= \vec{v}_1 + 2\vec{v}_2 + \vec{v}_3 + \vec{v}_4.
\end{align*}
\]

The p-span of these vectors is not a module, since it does not contain the vector \( 5\vec{v}_4 = (6766) \). The reason is that we cannot order the vectors so that they satisfy the definition of a p-generator sequence.

**Example 5:** The following set of vectors over \( Z_9 \) forms a p-generator sequence:

\[
\begin{align*}
\vec{v}_1 &= (0101) \\
\vec{v}_2 &= (2500) \\
\vec{v}_3 &= (5203) \\
\vec{v}_4 &= (3300) \\
3\vec{v}_1 &= 2\vec{v}_2 + \vec{v}_3 \\
3\vec{v}_2 &= 2\vec{v}_4 \\
3\vec{v}_3 &= 2\vec{v}_4 \\
3\vec{v}_4 &= 0.
\end{align*}
\]

Let us see how to obtain a p-linear combination equivalent to

\[
\vec{u} = 7\vec{v}_1 + 4\vec{v}_2 + \vec{v}_3 + 2\vec{v}_4 = (1801)
\]

Let us give an intuitive justification for the definition of p-generator sequences. The definition is motivated by computational considerations. If the vectors can be ordered as required in the definition, computations will proceed in an orderly fashion along the ordering; this is proven rigorously in Theorem 6.2. Otherwise, computations get “entangled” in loops. In fact, we conjecture that if the vectors in \( V \) cannot be so ordered, then either p-linear combinations of \( V \) do not generate a module, or the two notions of dependence do not turn out to be equivalent (see Theorem 6.3); however, we expect the proof to be quite involved. Examples 4 and 5 illustrate this point.

**Lemma 6.1:** Let \( V \) be a p-generator sequence, with \( |V| = k \). Let

\[
\vec{v} = \sum_{i=1}^{k} a_i \vec{v}_i
\]

be any linear combination of vectors in \( V \), and let \( \vec{v}_1 \) be the earliest vector used in this linear combination. Then \( \vec{v} \) can be expressed as a p-linear combination of \( \vec{v}_1 \) and later vectors of \( V \).

**Proof:** The coefficients occurring in any linear or p-linear combination can be written as a \( k \)-dimensional vector. Let \( (b_1, \ldots, b_k) \) and \( (c_1, \ldots, c_k) \) be two such vectors, and let \( b_i \) and \( c_j \) be their first nonzero coefficients. We will say that \( (b_1, \ldots, b_k) \) is lexicographically larger than \( (c_1, \ldots, c_k) \) if either \( i < j \), or \( i = j \) and \( b_i > c_i \).

Now consider the coefficient vector \( (a_1, \ldots, a_k) \). If all coefficients are in the range \( \{0, 1, \ldots, p - 1\} \), then we are done. Otherwise, let \( a_j \) be the first coefficient that is not less than \( p \). Let \( a_j = ap + b \), where \( 0 \leq b \leq p - 1 \). Write \( ap\vec{v}_j \) using the canonical p-linear combination for \( \vec{v}_j \). This uses vectors occurring later than \( \vec{v}_j \). Substituting, we will get a vector equivalent to \( (a_1, \ldots, a_k) \), which is the same in the first \( j - 1 \) places, and has \( b \) in the \( j \)th place. So this vector is lexicographically smaller than \( (a_1, \ldots, a_k) \). This process can be continued until we get an equivalent p-linear combination, since the process clearly terminates. The final vector must have zero coefficients in the first \( l - 1 \) places.

**Theorem 6.2:** If \( V \) is a p-generator sequence, then

\[
p-span(V) = span(V).
\]
Proof: Clearly \( \text{p-span}(V) \subseteq \text{span}(V) \). Since by Lemma 6.1 every vector in \( \text{span}(V) \) can also be expressed as a \( p \)-linear combination of vectors in \( V \), the reverse inclusion also holds.

**Theorem 6.3:** Let \( V \) be a \( p \)-generator sequence. With respect to \( p \)-linear combinations over \( V \), the two notions of linear dependence are equivalent; i.e., there is a nontrivial \( p \)-linear combination of vectors in \( V \) that is equal to 0 iff there is a vector in \( V \) that can be expressed as a \( p \)-linear combination of the remaining vectors in \( V \).

**Proof:** First suppose that there is a nontrivial \( p \)-linear combination

\[
\sum_{i=1}^{k} a_i \vec{v}_i = 0.
\]

Let \( \vec{v}_i \) be the earliest vector used by this \( p \)-linear combination. Now we have

\[
a_i \vec{v}_i = \left( \sum_{i=1}^{k} a_i \vec{v}_i \right) = \sum_{i=1}^{k} \left( p^\alpha - 1 \right) a_i \vec{v}_i.
\]

Since \( a_i \) is a unit in \( Z_{p\alpha} \), we get

\[
\vec{v}_i = (a_i)^{-1} \sum_{i=1}^{k} \left( p^\alpha - 1 \right) a_i \vec{v}_i.
\]

Now, by Lemma 6.1, the linear combination on the right-hand side (r.h.s.) can be expressed as a \( p \)-linear combination which does not use \( \vec{v}_i \). So, \( \vec{v}_i \) has been expressed as a \( p \)-linear combination of the remaining vectors.

On the other hand, suppose

\[
\vec{v}_l = \sum_{i \neq l} a_i \vec{v}_i.
\]

Let \( \vec{v}_j \) be the earliest vector used in the r.h.s. There are two cases:

**Case 1:** \( l < j \). By Lemma 6.1

\[
\vec{v}_l - \left( \sum_{i \neq l} a_i \vec{v}_i \right)
\]

can be expressed as a \( p \)-linear combination using vectors later than \( \vec{v}_l \) only. Hence

\[
\vec{v}_l - \left( \sum_{i \neq l} a_i \vec{v}_i \right)
\]

can be written as a nontrivial \( p \)-linear combination that is equal to 0.

**Case 2:** \( l > j \). By Lemma 6.1, \( -\vec{v}_l \) can be expressed as a \( p \)-linear combination that does not use \( \vec{v}_j \). Hence

\[
\sum_{i \neq l} a_i \vec{v}_i - \vec{v}_l
\]

can be written as a nontrivial \( p \)-linear combination that is equal to 0.

We will say that a \( p \)-generator sequence \( V \) is **\( p \)-linearly independent** if there is no nontrivial \( p \)-linear combination of its vectors that is equal to 0. A \( p \)-linearly independent \( p \)-generator sequence will be called a **\( p \)-basis**. Clearly, the \( p \)-linear combinations of the elements of a \( p \)-basis \( V \) uniquely generate the elements of the \( p \)-span \( \text{span}(V) \). So, if \( |V| = k \), the module has \( p^k \) elements. We will define the **\( p \)-dimension** of this module to be \( k \).

**Remark:** The notions of \( p \)-dimension and \( p \)-generator sequence of a module over \( Z_{p\alpha} \) are exactly the notions of composition length and generating system along a composition chain of a module of commutative ring theory (see [20]). The reason for our choice of terminology is that it is more suggestive of properties of vector spaces that we are attempting to extend to modules over \( Z_{p\alpha} \).

**Lemma 6.4:** Every module over \( Z_{p\alpha} \) has a \( p \)-generator sequence.

**Proof:** Let \( U = \{ \vec{v}_1, \ldots, \vec{v}_k \} \) be a generating set for the module in the ordinary sense. Let \( V \) be the ordered sequence consisting of multiples of these vectors by \( p^i, 0 \leq i \leq \alpha - 1 \), i.e.,

\[
V = (\vec{v}_1, p\vec{v}_1, \ldots, p^{\alpha-1}\vec{v}_1, \ldots, \vec{v}_k, p\vec{v}_k, \ldots, p^{\alpha-1}\vec{v}_k).
\]

Then, clearly, \( V \) is a \( p \)-generator sequence with \( p \)-span \( \text{span}(V) \). Notice that if any of the vectors in \( V \) is 0, it can be omitted.

Our next goal is to show that every module over \( Z_{p\alpha} \) has a \( p \)-basis. We will do this by adapting Gaussian elimination to this setting.

Let us first recall the process of Gaussian elimination when performed on vectors from a vector space. Let \( V = \{ \vec{v}_1, \ldots, \vec{v}_k \} \) be the generator set for a subspace of \( F^n \), the \( n \)-dimensional vector space over field \( F \). The process of Gaussian elimination is based on the following fact: Let

\[
\vec{v} = \sum_{i=1}^{k} a_i \vec{v}_i
\]

be a linear combination of the vectors in \( V \). Then, for any vector \( \vec{v}_i \) that is used by this linear combination, \( V + \vec{v} - \vec{v}_i \) generates the same subspace as \( V \). Using this principle, Gaussian elimination starts with an arbitrary generator set for a subspace, and brings it into “row echelon” form, i.e., all nonzero vectors have distinct starting points, and are sorted by starting point, with the 0 vectors being listed last. Now, the nonzero vectors are linearly independent, and form a basis for the subspace.

Carrying out this process is somewhat more involved for \( p \)-generator sequences.

**Lemma 6.5:** Let \( V = (\vec{v}_1, \ldots, \vec{v}_k) \) be a \( p \)-generator sequence, and let

\[
\vec{v} = \sum_{i=1}^{k} a_i \vec{v}_i
\]

be a \( p \)-linear combination of its vectors. Let \( \vec{v}_l \) be the earliest vector in this ordering that is used by the \( p \)-linear combination, and let \( U \) be obtained by replacing \( \vec{v}_l \) by \( \vec{v} \) in the ordered set \( V \). Then, \( U \) is also a \( p \)-generator sequence with the same span as \( V \).
Therefore, corresponding to any linear combination of the vectors of $V$ there is an equivalent linear combination of the vectors of $U$ and vice versa. Hence, $U$ has the same span as $V$.

Next we show that $U$ is a $p$-generator sequence; i.e., for each vector $v_i \in U$, $p v_i$ can be expressed as a $p$-linear combination of vectors later than $v_i$ in $U$. This is clearly true for $j > l$. For the vector $v$

$$p v = p v_i + \sum_{i \neq l} p a_i v_i.$$ 

Using Lemma 6.1, and the fact that there is a $p$-linear combination for $p v_i$ using vectors later than $l$, the r.h.s. can be expressed as a $p$-linear combination of vectors later than $l$. Finally, consider $j < l$. If the canonical $p$-linear combination for $p v_i$ in $V$ does not use $v_i$, we will simply use this same $p$-linear combination. Otherwise, we will substitute for $v_i$ using the first equation given above, and use Lemma 6.1 to obtain a $p$-linear combination that uses vectors of $U$ later than $v_i$.

**Corollary 6.6:** Let $V = \{v_1, \ldots, v_s\}$ be a $p$-generator sequence. Let $\bar{v} = v_i + a v_j$, where $i < j$ and $a \in \mathbb{Z}_p$. Then, replacing $v_i$ by $\bar{v}$ in $V$ gives an equivalent $p$-generator sequence.

**Proof:** The proof follows by observing that $a \bar{v}$ can be written as a $p$-linear combination of $v_j$ and later vectors of $V$.

We say that a $p$-generator sequence $V$ is proper if for each pair of nonzero vectors $v, \bar{v} \in V$, if $v$ and $\bar{v}$ have the same starting point, then their starting elements are not associates.

**Lemma 6.7:** Every submodule of $Z_{p^n}$ has a proper $p$-generator sequence.

**Proof:** Let $V$ be a $p$-generator sequence that is not proper. Say that vectors $v$ and $\bar{v}$ are in conflict if they have the same starting point, and their starting elements are associates. Among all conflicting pairs having the earliest starting point, pick a pair whose starting elements have the highest order. Let $v_i$ and $v_j$ be this pair, with $i > j$. Now, we can find $a \in \mathbb{Z}_p$ such that adding $a v_j$ to $v_i$ zeros out the starting element of $v_i$; i.e., $v = v_i + a v_j$ has a later starting point than $v_i$. By Corollary 6.6, replacing $v_i$ by $v_j$ in $V$ gives an equivalent $p$-generator sequence. Clearly, this process must terminate and yield a proper $p$-generator sequence.

We will say that a proper $p$-generator sequence $V = \{v_1, \ldots, v_k\}$ is in row echelon form if for $1 \leq i < j \leq k$ either:

1) $v_i$ has an earlier starting point than $v_j$, or 
2) $v_i$ and $v_j$ have the same starting point, and the starting element of $v_i$ has higher order than the starting element of $v_j$.

**Lemma 6.8:** Let $V = \{v_1, \ldots, v_s\}$ be a $p$-generator sequence in row echelon form. If $v_i$ is nonzero, then a $p$-linear combination for $p v_i$ cannot use any vector $v_j$ with $j < i$.

**Proof:** Suppose otherwise, and let $v_i$ be the earliest vector used. $V$ has at most one vector with a given starting point and order of starting element. Therefore, the remaining vectors used in the $p$-linear combination cannot zero-out the starting element of $v_i$. Then this $p$-linear combination will either start before $v_i$, or will start at the same point as $v_i$ but with an element of higher order than the starting element of $v_i$. In either case we get a contradiction.

**Corollary 6.9:** Let $V$ be a proper $p$-generator sequence. Then, permuting its vectors so they are in row echelon form gives an equivalent $p$-generator sequence.

**Lemma 6.10:** The nonzero vectors of a $p$-generator sequence in row echelon form are $p$-linearly independent.

**Proof:** The proof is along the same lines as in Lemma 6.8. Consider any nontrivial $p$-linear combination of the vectors. Then the starting element of the earliest vector used cannot be canceled out by the remaining vectors. Hence, no nontrivial $p$-linear combination of the vectors can be equal to 0.

**Theorem 6.11:** Every submodule of $Z_{p^n}$ has a $p$-basis.

Finally, we give below a Gaussian elimination procedure that starts with an arbitrary $p$-generator sequence for a submodule of $Z_{p^n}$, and finds a $p$-generator sequence in row echelon form. This procedure is designed along the lines of the usual Gaussian elimination procedure for obtaining a basis in row echelon form for the field case; it simultaneously carries out the process in Lemma 6.7, together with the permutation of vectors given in Corollary 6.9.

**Algorithm (Gaussian Elimination):**

1) $S \leftarrow V$.
2) While there is a nonzero vector in $S$ do:
3) Find $S' \subseteq S$, the set of vectors of $S$ having the earliest starting point.
4) Find $S'' \subseteq S'$, the set of vectors of $S'$ having the highest order starting element.
5) Pick the last vector $v \in S''$, output it, and set $S \leftarrow S - \{v\}$.
6) For each remaining $v \in S''$, replace $v$ in $S$ by $(a + a v)$, where $a \in \mathbb{Z}_{p^n}$ is such that $(a + a v)$ starts later than $v$.

7) end.

**Theorem 6.12:** The Gaussian elimination algorithm given above starts with an arbitrary $p$-generator sequence for a submodule and finds a proper $p$-generator sequence in row echelon form. Its running time is bounded by $O(k^2 n)$ operations over $Z_{p^n}$, where $k$ is the number of vectors in the $p$-generator sequence.

**Example 6:** Consider the code over $Z_8$ generated by:

\[
\begin{bmatrix}
1212 \\
2042 \\
0044
\end{bmatrix}
\]

A $p$-generator sequence for it is given below. Since there are two rows starting with a 2 in column one, we add three times row 3 to row 2. Next, we add row 6 to row 5.
discarding duplicate rows, we get a $p$-basis.

$$\begin{pmatrix} 1212 \\ 2042 \\ 2424 \\ 0044 \\ 4004 \end{pmatrix} \rightarrow \begin{pmatrix} 1212 \\ 0426 \\ 2424 \\ 0044 \\ 4004 \end{pmatrix} \rightarrow \begin{pmatrix} 1212 \\ 0426 \\ 2424 \\ 0044 \\ 4004 \end{pmatrix} \rightarrow \begin{pmatrix} 1212 \\ 0426 \\ 2424 \\ 0044 \\ 4004 \end{pmatrix}$$

VII. MINIMAL TRELLISES FOR CODES OVER RINGS $\mathbb{Z}_{p^n}$

In this section, we will present a polynomial-time algorithm for constructing a minimal trellis for a linear code over a ring $\mathbb{Z}_{p^n}$, given a generator matrix for it. Let us first give a natural generalization of the notion of a two-way proper matrix over a field.

A $p$-generator sequence $V$ will be said to be two-way proper if:

1) for each pair of vectors $\vec{u}, \vec{v} \in V$, if $\vec{u}$ and $\vec{v}$ start at the same point, then their ending elements are not associates, and
2) for each pair of vectors $\vec{u}, \vec{v} \in V$, if $\vec{u}$ and $\vec{v}$ end at the same point, then their starting elements are not associates.

Below we give an algorithm that starts with a proper $p$-generator sequence in row echelon form, $V$, and finds a two-way proper $p$-generator sequence having the same span.

**Algorithm (Two-Way Proper $p$-Generator Sequence):**

1) $S \leftarrow V$.
2) While $S$ is not two-way proper do:
3) Find $S' \subseteq S$, with $|S'| > 1$, vectors having the latest ending point, and moreover such that their ending elements are associates; 4) Let $\vec{u}$ be the last vector in $S'$.
5) For each remaining $\vec{u} \in S'$, replace $\vec{u}$ in $S$ by $(\vec{u} + a\vec{v})$, where $a \in \mathbb{Z}_{p^n}$ is such that $(\vec{u} + a\vec{v})$ ends earlier than $\vec{u}$.
6) end.

**Lemma 7.1:** The algorithm given above starts with a proper $p$-generator sequence in row echelon form, and finds a two-way proper $p$-generator sequence with the same span. Its running time is bounded by $O(k^2n)$ operations over $\mathbb{Z}_{p^n}$, where $k$ is the number of vectors in the $p$-generator sequence.

**Example 7:** The $p$-basis obtained in Example 6 is not two-way proper. We add row 2 to row 1 and row 4 to row 3 to obtain the final two-way proper $p$-basis:

$$\begin{pmatrix} 1630 \\ 0426 \\ 2460 \\ 0044 \\ 4040 \end{pmatrix}$$

Starting with a two-way proper $p$-generator sequence $V$, our trellis construction algorithm then has the same overall structure as the field case. The trellis for a single vector $\vec{v}$ of $V$ is required to generate the $p$ codewords that can be generated as $p$-linear combinations of this vector, i.e., $\{0, \vec{v}, 2\vec{v}, \ldots, (p - 1)\vec{v}\}$. This trellis is similar to the trellis for a single vector in the field case: it has a $p$-way fork at the time at which $\vec{v}$ starts, and a $p$-way collapse at the time at which $\vec{v}$ ends. The $p$ out-transitions will be labeled with 0 and the associates of the starting element of $\vec{v}$, and the $p$ in-transitions will be labeled with 0 and the associates of the ending element of $\vec{v}$.

We will next show that the product of the trellises for the vectors of $V$ is two-way proper and hence minimal.

Let $a_1, \ldots, a_k \in \mathbb{Z}_{p^n}$. For notational convenience, it will be useful to regard these elements as one-dimensional vectors belonging to the module $\mathbb{Z}_{p^n}$. We can then talk about all $p$-linear combinations of these elements.

**Lemma 7.2:** Let $a_1, \ldots, a_k \in \mathbb{Z}_{p^n}$ be a set of elements that are pairwise nonassociates. Then their $p$-linear combinations are distinct elements of $\mathbb{Z}_{p^n}$.

**Proof:** Notice that if we start with any $\alpha$ elements of $\mathbb{Z}_{p^n}$ that are pairwise nonassociates, they will form a $p$-basis for the one-dimensional module $\mathbb{Z}_{p^n}$. Hence the $p$-linear combinations of $a_1, \ldots, a_k$ will generate distinct elements in $\mathbb{Z}_{p^n}$.

It is easy to see that the converse of the statement in Lemma 7.2 is not true, i.e., one can construct an example in which even though $a_1, \ldots, a_k$ are not pairwise nonassociates, their $p$-linear combinations may still generate distinct elements of $\mathbb{Z}_{p^n}$. Yet the following holds:

**Theorem 7.3:** Let $V$ be a $p$-basis for a submodule of $\mathbb{Z}_{p^n}$. Then, the product of trellises for the vectors of $V$ is a minimal trellis for the submodule generated by $V$ if $V$ is two-way proper.

**Proof:** Suppose $V$ is two-way proper. Let us show that the product trellis will be forward proper; the proof that it is backward proper is similar. If $V$ has $k_i$ vectors, say $V_i$, that start at time $i$, then any state $s$ in the product trellis at time $i - 1$ will have $p^{k_i}$ out-transitions. Since the starting elements of the vectors in $V_i$ are nonassociates, by Lemma 7.2 the $p$-linear combinations of these vectors will all start with distinct elements. The set of symbols on the out-transitions of $s$ consist of some element $a \in \mathbb{Z}_{p^n}$ added to these distinct elements, so $s$ is forward proper.

Conversely, suppose $V$ is not two-way proper. Suppose there are two vectors $\vec{u}$ and $\vec{v}$ starting at index $i$, so that their starting elements are associates. (The proof when $V$ has two vectors whose ending points are the same, and the ending elements are associates is similar.)

Now there are units $b, c \in \mathbb{Z}_{p^n}$ such that $b\vec{u} + c\vec{v}$ starts at a later index than $i$. Let $V_i$ be the set of vectors of $V$ that start at index $i$. Using the fact that $V$ is a $p$-generator sequence, it follows that $b\vec{u} + c\vec{v}$ can be written as a $p$-linear combination of vectors in $V_i$ which is clearly nontrivial. In addition, the trivial $p$-linear combination of the vectors in $V_i$ also gives a vector that is 0 at time $i$.

Finally, let $s$ and $a$ be as defined above. Now there are two $p$-linear combinations of vectors in $V_i$ that give out-transitions with symbol $a$ from state $s$. Therefore, the product trellis is not two-way proper, and hence it is not minimal.
An information vector \( v' \in (C,') \) generates the codeword \( v'9 \), and the set of all codewords constructed in this manner is \( C \).

**Lemma 8.2:** Given a generator matrix \( \Psi \) for \( C \), we can obtain a \( km \times mn \) generator matrix \( A \) over \( \text{GF}(p) \) for \( S \).

**Proof:** As shown in [1], an endomorphism \( \psi : C_p^m \rightarrow C_p^m \) may be viewed as an \( m \times m \) matrix \( M_\psi \) over \( C_p^m \). View an element \( a \in C_p^m \) as an \( m \)-dimensional row vector \( a' \) with entries from \( C_p^m \). Then \( aM_\psi = \psi(a) \).

Thus \( \Psi \) may be viewed as a \( k \times n \) matrix whose elements are \( m \times m \) matrices over \( C_p^m \). Intuitively, the \( km \times mn \) matrix \( A \) is obtained by simply "removing the demarcations" of the element matrices of \( \Psi \). Formally, for \( 1 \leq i \leq km \) and \( 1 \leq j \leq mn \), divide \( i \) and \( j \) by \( m \) to obtain quotients and remainders \( q_i, q_j \) and \( r_i, r_j \), respectively. Now, let the \((i, j)\)th entry of \( A \) be the \((r_i, r_j)\)th entry of the matrix corresponding to \( \psi_{q_i, q_j} \), i.e.,

\[
A[i, j] = M_{\psi_{q_i, q_j}}[r_i, r_j].
\]

Clearly, \( A \) is a generator matrix for the code \( S \).

The algorithm for obtaining a minimal trellis for \( C \) is as follows: First, obtain a minimal trellis \( T \) for the linear code \( S \). This trellis will have length \( mn \). Next, sectionalize this trellis by collapsing \( m \) successive branches into one branch to obtain a trellis \( T' \); \( T' \) has length \( n \), and the set of states at time \( i \) in \( T' \) is the same as the set of states at time \( ki \) in \( T \). States \( u \) and \( v \) at successive times in \( T' \) have a transition iff there is a path from \( u \) to \( v \) in \( T \). So, each such path gives a symbol from the group \( C \) (of course, if there are multiple labels on transitions of \( T \), we will get multiple symbols from the same path); these symbols constitute the set of labels on this transition. It is easy to see that if \( T \) is two-way proper, then so is \( T' \) (note that \( T' \) may be two-way proper even though \( T \) is not).

**Lemma 8.3:** \( T' \) is a minimal trellis for \( C \).

**Example 8:** Consider the code over \( Z_4 \) generated by the following two-way proper \( p \)-generator sequence:

\[
\begin{pmatrix}
1 & 2 & 3 & 0 \\
2 & 0 & 2 & 0 \\
0 & 2 & 2 & 2
\end{pmatrix}
\]

The trellises for the individual rows as well as the product trellis are shown in Fig. 4.

**VIII. MINIMAL TRELLISES FOR CODES OVER FINITE ABELIAN GROUPS**

We will first extend our construction algorithm to codes over elementary abelian groups; this will illustrate in a simpler setting the main ideas in the extension to codes over arbitrary finite abelian groups. An elementary abelian group \( G \) is isomorphic to a direct product of cyclic \( p \)-groups, i.e., \( G \cong C_p^m \), where \( p \) is a prime.

**Lemma 8.1:** A length \( n \) group code \( C \) over \( G \cong C_p^m \) may be viewed as a linear code \( S \) of length \( mn \) over \( \text{GF}(p) \).

**Proof:** Using the natural isomorphism between \( C_p^m \) and the additive group of \( \text{GF}(p) \), we can view \( C \) as a length-\( mn \) code over \( \text{GF}(p) \), say \( S \). Since \( C \) is a group code, so is \( S \). Further, since multiplication in \( \text{GF}(p) \) is simply repeated addition, \( S \) is a linear code over \( \text{GF}(p) \).

Biglieri and Elia [1] have shown that \( C \) may be specified by a \( m \times n \) generator matrix \( \Psi \) whose entries are endomorphisms, \( \psi_{i,j} : C_p^m \rightarrow C_p^m \). An information vector \( \vec{v} \in (C_p^m)^k \) generates the codeword \( \vec{v}\Psi \), and the set of all codewords constructed in this manner is \( C \).

**Example 9:** For the elementary abelian group with four elements, \( C_2 \oplus C_2 = \{0, x \} \times \{0, y \} = \{0, x, y, xy\} \), consider the length-4 group code (see the bottom of this page). This is an MDS code that cannot be represented as a linear code over \( \text{GF}(4) \). Under the map

\[
0 \rightarrow (00), x \rightarrow (01), y \rightarrow (10), xy \rightarrow (11)
\]

this code becomes a length-8 linear code over \( \text{GF}(2) \). A generator matrix for the original group code is

\[
\begin{pmatrix}
0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 \\
1 & 0 & 1 & 0
\end{pmatrix}
\]

A generator matrix for the corresponding code over \( \text{GF}(2) \) is given below, along with a two-way proper generator matrix:

\[
\begin{pmatrix}
0 & 0 & 0 & 0 \\
x & 0 & xy & xy \\
y & 0 & y & y \\
xy & 0 & x & x
\end{pmatrix}
\]

\[
\begin{pmatrix}
0 & x & xy & y \\
x & 0 & x & y \\
y & 0 & y & xy \\
xy & y & xy & 0
\end{pmatrix}
\]
Fig. 5. The trellis for Example 9.

derived therefrom:
\[
\begin{pmatrix}
00011110 \\
01001111 \\
01101101 \\
11011011
\end{pmatrix}
\rightarrow
\begin{pmatrix}
11101100 \\
01111000 \\
00101001 \\
00011110
\end{pmatrix}
\]

Applying trellis construction procedures for linear codes, we obtain the trellis given in Fig. 5(top). Sectionalizing this trellis, and applying the reverse map from GF(2)^2 to C, we obtain the minimal trellis for the original code, shown in Fig. 5(bottom).

Finally, let us consider the case in which G is an arbitrary finite abelian group. Then, G is isomorphic to a direct product of cyclic groups, i.e.,
\[
G \cong \bigoplus_{i=1}^{l} C_{p_i^{a_i}}
\]

where \(p_1, \ldots, p_l\) are distinct primes (see [12]). As other authors have noticed (see [3]), this decomposition leads to a corresponding decomposition for codes.

**Lemma 8.4:** Let \(C\) be a length-\(n\) group code over a finite abelian group \(G\). Let \(G \cong G_1 \oplus G_2\), where the orders of \(G_1\) and \(G_2\) are relatively prime. Then there are length-\(n\) group codes \(C_1\) and \(C_2\) over \(G_1\) and \(G_2\), respectively, such that \(C \cong C_1 \oplus C_2\).

As a consequence of Lemma 8.4, we can decompose \(C\) into \(l\) codes \(C_1, \ldots, C_l\). We can then obtain minimal trellises for these \(l\) codes; however, we must view the labels in these trellises as if they were elements of \(G\), by using the natural injection maps. Now, using Theorem 4.1, it is easy to see that the product of these trellises is a minimal trellis for \(C\).

Hence, it is sufficient to consider the case of a code over a \(p\)-group \(G\), i.e., let
\[
G \cong C_{p^{a_1}} \oplus \cdots \oplus C_{p^{a_m}}
\]

where \(a_1 \leq \cdots \leq a_m\). The proofs of the next two lemmas are then identical to those of Lemmas 8.1 and 8.2.

**Lemma 8.5:** A length-\(n\) group code \(C\) over a \(p\)-group
\[
G = C_{p^{a_1}} \oplus \cdots \oplus C_{p^{a_m}}
\]

where \(a_1 \leq \cdots \leq a_m\) is equivalent to a linear code \(A\) of length \(mn\) over \(Z_{p^{a}}\), where \(a = a_m\).

Again, by the result of Biglieri and Elia [1], \(C\) may be specified by a \(k \times n\) generator matrix \(\Psi\) whose entries are endomorphisms \(\psi_{i,j}: G \rightarrow G\).
Lemma 8.6: Given a generator matrix $\Psi$ for $C$, we can obtain a $km \times mn$ generator matrix $A$ over $\mathbb{Z}_p$ for $S$.

Now, the structure of the trellis construction algorithm is similar to that for the elementary abelian case. We can obtain a generator matrix for the code over $\mathbb{Z}_p$ from that for $C$, and construct a minimal trellis for it. Then sectionalizing this trellis will give a minimal trellis for $C$. Hence, we get:

Theorem 8.7: There is an $O(k^2n + s)$ time algorithm that, given a generator matrix for a group code $C$ over an abelian group, constructs a minimal trellis for $C$. This algorithm also computes $s$ in $O(k^2n)$ time.

We illustrate the core ideas involved in the extension to the case of general finite abelian groups below:

Example 10: Consider the length-3 group code over $C_2 \times C_4$ specified by the following generator matrix:
\[
\begin{pmatrix}
1 & 2 \\
1 & 3 \\
0 & 2 \\
1 & 1
\end{pmatrix} \times
\begin{pmatrix}
1 & 0 \\
0 & 1 \\
0 & 0 \\
0 & 0
\end{pmatrix} =
\begin{pmatrix}
1 & 2 & 2 & 0 \\
1 & 3 & 2 & 0 \\
0 & 2 & 0 & 1 \\
1 & 1 & 0 & 0
\end{pmatrix}.
\]

Observe that the entries of the first column of every endomorphism matrix are over $\mathbb{Z}_2$, whereas the second column is over $\mathbb{Z}_4$. Since $\mathbb{Z}_2$ can be embedded into $\mathbb{Z}_4$ by the map $i \mapsto 2i, i \in \mathbb{Z}_2$, we can obtain an equivalent generator matrix over $\mathbb{Z}_4$ by using this map on the first column of each endomorphism matrix. Now, as in the previous example, we can view this code as a length-6 code over $\mathbb{Z}_4$. Applying our algorithm we obtain the following two-way proper generating matrix:
\[
\begin{pmatrix}
000 & 101 \\
012 & 100 \\
22 & 000 \\
000 & 220 \\
000 & 0220 \\
020 & 0200
\end{pmatrix}.
\]

Applying the trellis construction procedures for linear codes, we obtain the trellis given in Fig. 6(top). Sectionalizing this trellis and applying the reverse map, we obtain the minimal trellis for the original code, shown in Fig. 6(bottom).
IX. COMPUTING LOCAL DESCRIPTIONS OF MINIMAL TRELLISES

We will present efficient algorithms for the following two problems:

**Problem I:** Given states $s$ and $t$ at times $i$ and $i+1$, determine whether there is a transition from $s$ to $t$, and if so, the set of labels on this transition.

**Problem II:** Given state $s$ at time $i$, compute all states at time $i+1$ to which $s$ has transitions, and the sets of labels on these transitions. The problem of computing all states at time $i-1$ that have transitions into $s$ is analogous.

Kschischang and Sorokine [16] have given algorithms for these problems for linear codes over fields. Using the notion of $p$-linearity developed in Sections VI and VII, these algorithms extend to linear codes over rings $\mathbb{Z}_p^n$. In turn, using the concepts developed in Section VIII, these algorithms extend to codes over all finite abelian groups. Rather than directly presenting the algorithms for codes over finite abelian groups, we show below the natural progression of ideas; this will help us state the algorithms more clearly. For the field case, we extend to linear codes over fields. Using the notion of $p$-linearity, we show below the natural progression of ideas; this will help us state the algorithms more clearly. For the field case, we have modified the algorithms of Kschischang and Sorokine to start with a two-way proper generator matrix, rather than an arbitrary generator matrix.

A. The Field Case

Let $A$ be a $k \times n$ two-way proper generator matrix for a linear code $C$ over $\mathbb{GF}(q)$. For each $i$, $1 \leq i < n$, compute $a_i, b_i,$ and $c_i$ as follows:

- $a_i$ is the set of rows of $A$ that are zero in columns $i+1$ to $n$. Linear combinations of these rows of $A$ generate codewords in $C_{i-}$.
- $b_i$ is the set of rows of $A$ that are zero in columns 1 to $i$. Linear combinations of these rows of $A$ generate codewords in $C_{i+}$.
- $c_i$ is the remaining set of rows of $A$. Linear combinations of these rows of $A$ generate coset representatives for $C/(C_{i-} \oplus C_{i+})$, and are therefore in one-to-one correspondence with the set of states at time $i$ in the minimal trellis for $C$. We will denote state $s$ at time $i$ by a $k$-dimensional information vector $v_s$ that is zero in the components specified by $a_i$ and $b_i$. Thus $v_sA$ is the coset representative for state $s$, and $v_sA + (C/(C_{i-} \oplus C_{i+}))$ is the set of codewords that it is responsible for.

A succinct representation of the minimal trellis for $C$ consists of $A$ together with $a_i, b_i,$ and $c_i$ for each $i$, $1 \leq i < n$. This information can be computed in $O(k^2n)$ time, and requires $O(kn)$ space. In the following, we will denote the $ith$ column of $A$ by $A_i$.

The algorithm for Problem I is now straightforward: there is a transition from $s$ to $t$ iff the sets of codewords that they are responsible for have nonempty intersection. This happens iff $v_s$ and $v_t$ are identical on the components specified by $c_i \cap c_{i+1}$. If so, the set of codewords that use this transition are

$$\bigcup_u \{uA\}$$

where $u$ ranges over all $k$-dimensional vectors that agree with $v_s$ on positions specified by $c_i$ and with $v_t$ on positions specified by $c_{i+1}$. Thus the set of labels on this transition are given by

$$\bigcup_u \{uA_{i+1}\}$$

where $u$ is as specified above.

This expression can be simplified considerably. There are two cases. If $b_i \cap a_{i+1} = \emptyset$, then the positions specified by $(c_i \cup c_{i+1})$, $A_{i+1}$ is zero. Then the label on the transition is given by $uA_{i+1}$, where from the vectors $u$ given above, we have picked one that is zero on positions $(c_i \cup c_{i+1})$. Otherwise, $b_i \cap a_{i+1}$ is a single row of $A$ which is zero everywhere except in column $i+1$. In this case, the set of labels on the transition is all of $\mathbb{GF}(q)$.

The algorithm is summarized below. It is easy to check that it runs in $O(k)$ time.

**Algorithm (Local Description, Problem I):**

1) If $v_s$ and $v_t$ are not identical on components specified by $c_i \cap c_{i+1}$, then there is no transition from $s$ to $t$.

2) If $b_i \cap a_{i+1} \neq \emptyset$, then the transition from $s$ to $t$ is labeled with $\mathbb{GF}(q)$.

3) Else, the transition from $s$ to $t$ is labeled with $uA_{i+1}$, where $u$ agrees with $v_s$ on positions specified by $c_i$, agrees with $v_t$ on positions specified by $c_{i+1}$, and is zero on the remaining positions.

4) End.

Next, we give an algorithm for Problem II. If $c_{i+1} - c_i = \emptyset$, then there is only one transition out of state $s$ which goes to a state $t$ determined by $v_t$, where $v_t$ agrees with $v_s$ on the positions specified by $c_{i+1}$ and is zero everywhere else. The set of labels on this transition can be computed as above.

Otherwise, $c_{i+1} - c_i$ is a single row of $A$. In this case, there are $q$ transitions out of $s$ into states defined by the following vectors: for each $e \in \mathbb{GF}(q)$, consider the vector that equals $e$ in the position specified by $c_{i+1} - c_i$, agrees with $v_s$ on positions specified by $c_i \cap c_{i+1}$, and is zero elsewhere. The label is computed as in Step 3) of the algorithm given above. This can be made more efficient by precomputing $v_sA_{i+1}$, and adding $ef$, where $f$ is the symbol in $A$ at row $c_{i+1} - c_i$ and column $i+1$, for each $e \in \mathbb{GF}(q)$. Clearly, this takes $O(k)$ time.

**Example II:** Consider the code over $\mathbb{Z}_3$ generated by the following matrix:

$$
\begin{pmatrix}
100 & 202 \\
011 & 000 \\
001 & 110 \\
001 & 110
\end{pmatrix}
$$

Then

$$a_3 = \{2\}, \ b_3 = \{4\}, \ c_3 = \{1, 3\}$$


and

\[ a_4 = \{2, 3\}, \quad b_4 = \emptyset, \quad c_4 = \{1, 4\}. \]

Since \( b_3 \cap a_4 = \emptyset \), there is only one label on transitions from time 3 to time 4. The label on the transition from state (1020) at time 3 to state (1001) at time 4 is given by multiplying (1021) with the fourth column of the matrix, giving 2. Since \( c_4 - c_3 \neq \emptyset \), each state at time 3 has transitions to three states at time 4.

### B. Extending to Rings \( \mathbb{Z}_{p^n} \)

The algorithm and proof are similar to the field case, the main difference being that “linear combination” is replaced by “\( p \)-linear combination.” Let \( A \) be a two-way proper \( p \)-generator sequence over the ring \( \mathbb{Z}_{p^n} \). Let \( B \) be the submatrix of \( A \) given by columns \( km \times mn \) \( p \)-generator sequence over \( \mathbb{Z}_{p^n} \).

Let \( C \) be a length-\( n \) code over a finite abelian group \( G \). As in Section VIII, one can show that it is sufficient to consider the case

\[ G \simeq C_{p^{\alpha_1}} \oplus \cdots \oplus C_{p^{\alpha_m}} \]

where \( \alpha_1 \leq \cdots \leq \alpha_m \). Also, using Lemmas 8.5 and 8.6, the problem reduces to the case of rings \( \mathbb{Z}_{p^n} \). Let \( A \) be the \( km \times mn \) \( p \)-generator sequence over \( \mathbb{Z}_{p^n} \). The main differences are: with respect to \( A \), \( s \), and \( t \) correspond to states at times \( im \) and \( (i+1)m \). So, for example, \( s \) has a transition to \( t \) iff \( v_s \) and \( v_t \) agree on components specified by \( c_{im} \cap c_{(i+1)m} \).

#### X. Transition Space Theorem

In this section, we will present the Transition Space Theorem. This theorem helps define a succinct representation for minimal trellises for group codes, using which local descriptions of the trellis can be efficiently computed. However, unlike the succinct representation given for group codes over abelian groups in Section IX, in general this representation will not be of polynomial size, and so is less useful. Perhaps more importantly, the Transition Space Theorem gives algebraic structural properties of transitions in a minimal trellis for a group code. This theorem is derived as a corollary of the State Space Theorem of Forney and Trott [9], and can be viewed as a complementary theorem: the State Space Theorem characterizes states in a minimal trellis, while the Transition Space Theorem characterizes transitions.

Let \( \Sigma_k \) and \( \sigma_k \) denote the set of states and the state map at time \( k \), \( \sigma_k : C \rightarrow \Sigma_k \). The set of transitions from time \( k \) to \( k+1 \) is called the branch space and is denoted by \( \Sigma_{k,k+1} \) in [9]. Let \( \sigma_k \times \sigma_{k+1} \) be the Cartesian product of homomorphisms \( \sigma_k \) and \( \sigma_{k+1} \)

\[ \sigma_k \times \sigma_{k+1} : C \rightarrow \Sigma_k \times \Sigma_{k+1}. \]

Then

\[ \Sigma_{k,k+1} = [\sigma_k \times \sigma_{k+1}](C). \]

For conciseness, let us denote \( \Sigma_{k,k+1} \) by \( S^{(k)} \).

We will view \( S^{(k)} \) as a length-two group code. Then, at time 1, the codeword parts are

\[ P_- (S^{(k)}) = \Sigma_k = \frac{C}{C_k - \oplus C_{k+1}} \]

and the codeword futures are

\[ P_+ (S^{(k)}) = \Sigma_{k+1} = \frac{C}{C_{(k+1)} - \oplus C_{(k+1)+}}. \]
Denote the past subcode at time 1 by \( S(k)_- \), and the future subcode by \( S(k)_+ \), i.e.,
\[
S(k)_- = \{(a,0) | (a,0) \in S(k)\}
\]
\[
S(k)_+ = \{(0,b) | (0,b) \in S(k)\}.
\]

Then, by applying the State Space Theorem to this length-two code, we get:

**Theorem 10.1 (Transition Space Theorem):**
\[
\frac{S(k)}{S(k)_- \times S(k)_+} \cong \frac{\Sigma_k}{\Sigma_{k+1}} = \frac{\Sigma_{k+1}}{\Sigma_k}.
\]

Let \( A \) and \( B \) be sets of states at times \( k \) and \( k+1 \) in the minimal trellis \( T \) for the code \( C \). We will say that there is a **clique of transitions** from \( A \) to \( B \) if every state in \( A \) has a transition to every state in \( B \) in \( T \). The Transition Space Theorem shows that the branch space \( S(k) \) is partitioned into disjoint cliques of transitions, each corresponding to a coset of \( S(k)_- \times S(k)_+ \) in \( S(k) \). In particular, the set of transitions in \( S(k)_- \times S(k)_+ \) will be called the **zero clique**. See Fig. 7 for a good illustration of this fact.

The Transition Space Theorem enables us to give a succinct representation of the minimal trellis for a group code: for each time \( k \), store the zero clique of transitions and quotient group
\[
\frac{S(k)}{S(k)_- \times S(k)_+}.
\]

Clearly, this information is sufficient to compute the local description of the trellis.

**XI. LATTICE CODES**

Our algorithm can be used for constructing minimal trellises for lattices. Lattice codes constitute an important class of coset codes. An excellent treatment of lattice codes and trellises for lattice codes can be found in [6].

A real \( N \)-dimensional lattice is said to be **rectangular** if it has a generator matrix which is diagonal [2]. A lattice has a finite-state trellis diagram with respect to a given set of coordinates if and only if it contains a sublattice that is rectangular with respect to the same set of coordinates. Let \( M = \text{diag}(a_1, a_2, \cdots, a_N) \) be a generator matrix of an \( N \)-dimensional rectangular lattice, say \( \Lambda \), and let \( \Lambda_R \) be the maximal rectangular sublattice in \( \Lambda \). Then \( \Lambda_R \) has "trivial dynamics" and is the "nondynamical component" of \( \Lambda \) [9]. The quotient \( \Lambda/\Lambda_R \) is a finite abelian group and the techniques presented in our work can be used to study the dynamical structure of this group. We illustrate this with the following example:

**Example 12:** Let the lattice \( \Lambda \) be generated by
\[
M = \begin{bmatrix}
1 & 2 & 0 & 0 \\
1 & -2 & 0 & 0 \\
0 & 2 & -3 & 0 \\
0 & 0 & 3 & -4
\end{bmatrix}
\]

The maximal rectangular sublattice \( \Lambda_R \) is generated by \( \text{diag}(2, 4, 6, 8) \) and the quotient \( \Lambda/\Lambda_R \) is isomorphic to a subgroup of \( \mathbb{Z}_2 \times \mathbb{Z}_4 \times \mathbb{Z}_6 \times \mathbb{Z}_8 \). The generator matrix for this subgroup can be obtained by taking the \( i \)th column modulo \( a_i \) and discarding redundant rows and is shown below.
\[
\begin{bmatrix}
1 & 2 & 0 & 0 \\
0 & 2 & 3 & 0 \\
0 & 0 & 3 & 4
\end{bmatrix}
\]

The corresponding trellis diagram is shown in Fig. 8(a). The trellis diagram for \( \Lambda \) can be obtained by taking the product of this trellis with the trellis for \( \Lambda_R \) shown in Fig. 8(b).
Fig. 8. The trellis for Example 12.

XII. DISCUSSION

A natural first step in extending our work to codes over nonabelian groups is to consider group codes over semidirect product groups, for example dihedral groups. Group codes over such groups that are obtainable using multilevel constructions have been characterized in [10], and a theorem (Theorem 3) on trellis construction has been proposed. We expect that if we first obtain minimal trellises for the component codes of the group code using our algorithm and then take the product of the resulting trellises, we will get the minimal trellis for the block group codes of [10].

A further extension to group codes over arbitrary finite nonabelian groups seems difficult at present since we do not know of generator-matrix descriptions for such codes. In particular, the set of endomorphisms of a nonabelian group in general do not form a ring.

Another research direction worth investigating is the special structure of minimal trellises for group codes to obtain faster decoding algorithms. Towards this end, the State Space Theorem of Forney and Trott [9], together with the Transition Space Theorem and the succinct representation of minimal trellises presented here may prove useful.

ACKNOWLEDGMENT

The authors wish to thank an anonymous reviewer for suggesting several important improvements, including pointing out the application to lattices, and suggesting Example 12.

REFERENCES