

The Trellis Structure of Maximal Fixed-Cost Codes

Frank R. Kschischang

Department of Electrical and Computer Engineering
University of Toronto, Toronto, Ontario, Canada M5S 1A4

July 18, 1996

⁰This paper will appear in *IEEE Trans. on Inform. Theory*, vol. 42, no. 6, pp. xxxx-xxxx, Nov. 1996. Due to the editorial process, there may be slight discrepancies between this version and the published one. The latter should be regarded as the authoritative version. This paper is available over the internet at <http://www.comm.utoronto.ca/frank/>

The Trellis Structure of Maximal Fixed-Cost Codes

Frank R. Kschischang

Department of Electrical and Computer Engineering
University of Toronto, Toronto, Ontario, Canada M5S 1A4
`frank@comm.utoronto.ca`

July 18, 1996

Abstract—We show that the family of maximal fixed-cost (MFC) codes, with codeword costs defined in a right-cancellative semigroup, are rectangular, and hence admit biproper trellis presentations. Among all possible trellis presentations for a rectangular code, biproper trellises minimize a wide variety of complexity measures, including the Viterbi decoding complexity. Examples of MFC codes include such “nonlinear” codes as permutation codes, shells of constant norm in the integer lattice, and linear codes over a finite field. The intersection of two rectangular codes is another rectangular code; therefore “nonlinear” codes such as lattice shells or words of constant weight in a linear code have biproper trellis presentations. We show that every rectangular code can be interpreted as an MFC code. Applications of these results include error detection, trellis-based indexing, and soft-decision decoding.

Keywords: maximal fixed-cost codes, rectangular codes, biproper trellises, minimal trellises, lattice codes, constant weight subcodes.

I Introduction

A biproper trellis presentation for a block code C is one in which both the set of edges incident *from* any trellis vertex and the set of edges incident *to* any trellis vertex are labelled distinctly. Not all codes have biproper trellis presentations [1]; however, when a code has such a trellis presentation, the fortunate circumstance arises in which the biproper trellis simultaneously minimizes the vertex count at each time index, the edge count in each trellis section, and the overall Viterbi decoding complexity (Theorem 4). Furthermore, the biproper trellis presentation is unique (up to graph isomorphism), the trellis is one-to-one, and subtrellises are also biproper. A code admits a biproper trellis presentation if and only if it is *rectangular*. In the language of dynamical systems theory [2, 3], every rectangular code has a well-defined and unique minimal state realization.

Trellises were introduced in the coding theory literature by Forney [4] as a means of describing the Viterbi algorithm for decoding convolutional codes. Bahl, *et al.* [5] showed that block codes, too, can be described by a trellis, and Wolf [6] proposed the use of the Viterbi algorithm for trellis-based soft-decision decoding of block codes. Massey [7] gave a graph-theoretic definition of a block trellis and an alternative construction for minimal trellises. Forney's celebrated paper [8] showed that group codes, including linear codes and lattices, have a well defined trellis structure. Forney's paper sparked a flurry of activity in this area (see, e.g., [9] for a list of references), including some of the papers included in this special issue. Much of this work (for example [2]), is motivated by the fact that group codes—and linear codes in particular—are rectangular and hence have a well-defined and unique (biproper) minimal trellis.

The purpose of this paper is to show that the codes in the wider class of *maximal fixed-cost codes* (MFC codes) are also rectangular. Such “nonlinear” codes as permutation codes and shells of the integer lattice in any dimension are MFC codes, as are, of course, ordinary linear codes over a finite field. Since the intersection of two rectangular codes is a rectangular code, the constant weight subcodes of a linear code and the fixed-norm shells in a lattice also have biproper trellis presentations.

This paper is organized as follows. In Section II we briefly review some of the relevant

background terminology, and describes a criterion for vertex mergeability in a trellis. In Section III we describe some of the important properties of biproper trellises, including minimality and uniqueness. Among all possible trellis presentations for a rectangular code, the biproper trellis minimizes a variety of trellis complexity measures, including edge count, vertex count, and Viterbi decoding complexity.

In Section IV, which contains the main results of this paper, we introduce the notion of a cost framework, and define an MFC code with respect to a given cost framework. We show not only that every MFC code is rectangular, but that every rectangular code is an MFC code with respect to some cost framework. Examples are given of MFC codes that include linear codes over a finite field, permutation codes, lattice shells of fixed norm, and constant-weight subcodes of a linear code.

In Section V we show that the minimal trellis for an MFC code can be constructed as a subtrellis of the complete cost trellis for a given cost framework, generalizing to MFC codes the trellis-construction method of [5, 6].

Finally, Section VI gives some concluding remarks, including a brief discussion of some of the applications of these results.

II Minimal Trellises

A Terminology

Throughout this paper, for integers a and b , we let $[a, b]$ denote the set $\{a, a + 1, \dots, b\}$, which is taken to be empty if $a > b$. The cardinality of a finite set X is denoted as $|X|$. The following terminology is fairly standard in graph theory.

An *edge-labeled digraph* is a triple (V, E, A) , where V is a set of vertices, A is a set (or “alphabet”) of edge labels, and $E \subset V \times A \times V$ is a set of directed edges. An edge $e = (v, a, v') \in E$ is said to have *initial vertex* $i(e) = v$, *final vertex* $f(e) = v'$, and *label* $\ell(e) = a$. An edge (v, a, v') is said to be *incident from* v and *incident to* v' . The set of edges incident to a vertex v is denoted $E_{\text{in}}(v)$ and the set of edges incident from a vertex v is denoted $E_{\text{out}}(v)$. The *in-degree* of v is $|E_{\text{in}}(v)|$ and the *out-degree* of v is $|E_{\text{out}}(v)|$.

We say that two edge-labeled digraphs $G = (V, E, A)$ and $G' = (V', E', A)$ are *isomorphic* and write $G \cong G'$ if there exists a bijection $g : V \rightarrow V'$ such that $(v_1, a, v_2) \in E$ if and only if $(g(v_1), a, g(v_2)) \in E'$.

A *path* p of length j from a vertex u to a vertex v in an edge-labeled digraph is a sequence $p = \{e_1, e_2, \dots, e_j\}$ of edges with $i(e_1) = u$, $f(e_j) = v$, and $f(e_i) = i(e_{i+1})$ for all $i \in [1, j - 1]$. If, for some i , an edge $e_i \in p$ is incident to or from a vertex v , then p is said to pass through v . A path $p = \{e_1, e_2, \dots, e_j\}$ has *label string* $\ell(p) \stackrel{\text{def}}{=} \ell(e_1)\ell(e_2)\cdots\ell(e_j)$. Associated with each vertex v is the *empty path* of length zero from v to itself. If p is an empty path, then $\ell(p) = \epsilon$ is the empty string. A *cycle* is a path of nonzero length from a vertex to itself.

We denote the *concatenation* of label strings x and y by juxtaposition. Of course $\epsilon x = x\epsilon = x$ for any label string x . If C is a collection of label strings, and $xy \in C$, we say x is followed by y in C , or y follows x in C . For sets of label strings X and Y , let $XY = \{xy : x \in X, y \in Y\}$. The set XY is said to have “product form.”

B Block Trellises

The following definition generalizes the definition of a trellis given in [1], which in turn was based on the trellis definitions of [7, 10].

Definition. A *trellis* is an edge-labeled digraph $T = (V, E, A)$ with the property that, given any two vertices $v, v' \in V$, if there is a path of length n from v to v' in T , then *every* path from v to v' in T has length n .

A sub-trellis is a subgraph of a trellis.

In a (block) trellis, it is usual to designate two special vertices: the *root*, r , and the *goal*, g . In such a trellis, a vertex v with the property that there is a path from r to v and also a path from v to g is said to be *essential* with respect to (r, g) . If every vertex is essential with respect to (r, g) , then the trellis is said to be *reduced* with respect to (r, g) . Let $T(r, g)$ denote the sub-trellis obtained from a trellis T by removing all vertices that are not essential with respect to (r, g) (and all edges incident to such vertices). Clearly

$T(r, g)$ is reduced with respect to (r, g) , and T itself is reduced with respect to (r, g) if and only if $T = T(r, g)$. In this paper we will deal mainly with reduced trellises, although in Section V we construct reduced trellises as sub-trellises of certain unreduced trellises.

The digraph obtained by reversing the direction of all edges in a trellis T is also a trellis, called the *reverse trellis* T^R . If T is reduced with respect to (r, g) , then T^R is reduced with respect to (g, r) .

By definition, the length of any path in $T(r, g)$ from r to a given trellis vertex v is the same for any path from r to v ; this number is called the *depth* of v , and is denoted $d(v)$. By convention, and since r is reached by the empty path from r , $d(r) = 0$. By definition, for a trellis of length n , $d(g) = n$. For $i \in [0, n]$, we let $V_i \stackrel{\text{def}}{=} \{v \in V : d(v) = i\}$ be the set of vertices at depth i in T .

Because the length of every path between two vertices is fixed, a trellis can have no cycles. In a reduced trellis (V, E, A) of length n , every edge e incident from a vertex at depth i , say, must be incident to a vertex at depth $i + 1$; i.e., for all $e \in E$, $d(f(e)) = d(i(e)) + 1$. For $i \in [1, n]$, we let $E_i \stackrel{\text{def}}{=} \bigcup_{v \in V_i} E_{\text{in}}(v)$ be the set of edges incident to the vertices at depth i . The set of edges E_i is sometimes referred to as the *i th trellis section*.

For a fixed vertex v in a reduced trellis $T(r, g)$, define

$$P(v) = \{\ell(p) : p \text{ is a path from } r \text{ to } v\}$$

and

$$F(v) = \{\ell(p) : p \text{ is a path from } v \text{ to } g\}$$

to be, respectively, the *past* and *future* of v . Then $P(r) = F(g) = \{\epsilon\}$, while $F(r) = P(g)$ is the set of label strings of all paths from the root to the goal. For a length n trellis $T(r, g)$ with label alphabet A , $F(r)$ is a block code of length n over A , usually denoted as $C(T)$. The code $C(T)$ is said to have *trellis presentation* T .

In a length n reduced trellis T with vertex set V , for every depth $d \in [0, n]$, we have

$$C(T) = \bigcup_{v \in V_d} P(v)F(v). \quad (1)$$

In words, a reduced trellis T provides a *cover* for $C(T)$ as a union of product-form subsets at each trellis depth. A given vertex v is said to cover the elements of the subcode $P(v)F(v)$.

Of course, a given codeword may, in general, be covered by more than one vertex at a given trellis depth.

It is convenient to extend this temporal “past/future” language to codes. For a length n code C over an alphabet A , and for $d \in [0, n]$, a codeword $c = a_1 a_2 \cdots a_n \in C$ is said to have *past* $P_d(c) = a_1 \cdots a_d$ and *future* $F_d(c) = a_{d+1} \cdots a_n$, where, by convention, $P_0(c) = F_n(c) = \epsilon$. At depth d , the code C has past $P_d(C) = \{P_d(c) : c \in C\}$ and future $F_d(C) = \{F_d(c) : c \in C\}$. Indeed, for a fixed d , $C \subset P_d(C)F_d(C)$, so C can be considered to be a code of length two over the alphabet $P_d(C) \cup F_d(C)$, a viewpoint introduced in [2].

Of course, the usual problem is not to find $C(T)$ given T , but rather, given C , to find a trellis presentation T with $C(T) = C$. In this paper, we consider only finite block codes, and, consequently, finite trellises. In so doing, we avoid certain difficulties (such as a potential lack of completeness [2, 3, 11, 12]) that arise with sequence codes.

Every block code C has a reduced trellis presentation, i.e., there exists a reduced trellis T with $C(T) = C$. For example, the “trivial trellis” [3] for a code C of length n has exactly $|C|$ “parallel” paths from r to g , in which, for $0 < i < n$, $|V_i| = |C|$ and $|E_{\text{in}}(v)| = |E_{\text{out}}(v)| = 1$ for all $v \in V_i$. In general, a code may have many trellis presentations. We shall be interested in “minimal” trellis presentations.

C Vertex Mergeability

A pair $\{v, v'\}$ of vertices in a reduced trellis T is said to be *mergeable* if $v \neq v'$ and

$$P(v)F(v') \cup P(v')F(v) \subset C(T).$$

In other words, a pair of trellis vertices is mergeable if concatenating the past of one with the future of the other produces no strings that are not codewords. If $\{v, v'\}$ is a mergeable pair, then necessarily $d(v) = d(v')$.

A pair of mergeable vertices in a trellis can be combined to yield a “smaller” trellis for the same code. Let $\{v, v'\}$ be a mergeable pair in a reduced trellis $T = (V, E, A)$ and introduce a vertex v^* not in V . Let $V' = (V \setminus \{v, v'\}) \cup \{v^*\}$ and define the map $m : V \rightarrow V'$ by $m(x) = x$ if $x \notin \{v, v'\}$ and $m(v) = m(v') = v^*$. Define $E' \subset V' \times A \times V'$ as the image

of E under the map which sends (x, a, y) to $(m(x), a, m(y))$. Let $T' = (V', E', A)$; then T' is smaller than T in the sense that $|V'| = |V| - 1$ and $|E'| \leq |E|$.

To see that $C(T') = C(T)$ we note that in T' , $P(v^*) = P(v) \cup P(v')$ and $F(v^*) = F(v) \cup F(v')$, hence

$$P(v^*)F(v^*) = P(v)F(v) \cup P(v)F(v') \cup P(v')F(v) \cup P(v')F(v'). \quad (2)$$

Combining (1) with (2) we observe that

$$\begin{aligned} C(T') &= \bigcup_{v \in V'_d} P(v)F(v) \\ &= P(v)F(v') \cup P(v')F(v) \cup \bigcup_{v \in V_d} P(v)F(v) \\ &= P(v)F(v') \cup P(v')F(v) \cup C(T) \\ &= C(T) \end{aligned}$$

where the last equality follows from the definition of mergeability. Thus, both T and T' are trellis presentations for the same code.

Remark: One might define a “local mergeability” property as follows. Define two trellis vertices v and v' in a reduced trellis to be *locally mergeable* if $v \neq v'$ and

$$P(v)F(v') \cup P(v')F(v) \subset P(v)F(v) \cup P(v')F(v').$$

In words, v and v' are locally mergeable if concatenating the past of one with the future of the other produces no strings not already covered by the two vertices separately. Equivalently, v and v' are locally mergeable if they are distinct and one of the following conditions holds: (a) $P(v) = P(v')$; or (b) $F(v) = F(v')$; or (c) $P(v) \subset P(v')$ and $F(v) \subset F(v')$; or (d) $P(v') \subset P(v)$ and $F(v') \subset F(v)$. Testing for local mergeability requires an examination only of the past and future of v and v' , and not of the complete set of codewords. Local mergeability implies mergeability, but the converse is not true. Fig. 1 shows a trellis with no locally mergeable vertices, but in which any pair of vertices at depth one is mergeable. This example shows that examining the past and future of two vertices without reference to the entire set of codewords is insufficient to determine mergeability of the two vertices.

(End of Remark.)

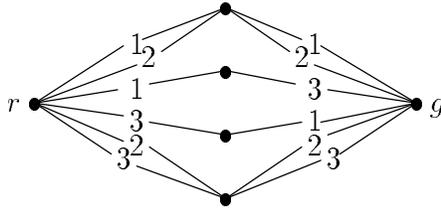


Figure 1: A nonminimal trellis with no locally mergeable vertices.

The notion of mergeability induces a partial ordering among the reduced trellises (or, more precisely, among the equivalence classes of isomorphic trellises) that represent a given code. We write $T' \preceq T$ if $C(T') = C(T)$ and T' is isomorphic to any trellis that can be obtained from T via a (possibly empty) sequence of vertex mergings. A reduced trellis T for which $T' \preceq T$ implies $T' \cong T$, i.e., an element that is minimal in this partial ordering, is said to be a *minimal* trellis. Clearly a reduced trellis is minimal if and only if it contains no mergeable pairs.

In general, a code may have more than one minimal trellis presentation. It is important to note that a minimal trellis may *not* minimize the number of vertices in the trellis. In other words, some minimal trellis presentations may be more economical (in terms of vertex count) than others. (See, for example, Fig. 2 of [10]). The reader should note that our use of the term “minimal” differs from some of the previous work on block code trellises (e.g., [10]), in which minimality is defined in terms of trellis vertex count, but our usage is consistent with notions of minimality in systems theory (e.g., [3, 12]).

III Codes with Unique Minimal Trellises

A Rectangular Codes

Recall that a code C has past $P_t(C)$ and future $F_t(C)$ at depth t . A code C of length $n > 1$ is said to be *rectangular* if, for all $t \in [1, n - 1]$,

$$\{ac, ad, bc\} \subset C \text{ implies } bd \in C \tag{3}$$

for all possible choices of $a, b \in P_t(C)$ and $c, d \in F_t(C)$.

We view (3) as a “rectangular closure” property that requires all possible rectangles of codewords in the past/future array [1] of a rectangular code to be “filled in” as in Fig. 2. As we shall see, rectangular codes are precisely those codes that admit a unique minimal trellis presentation.

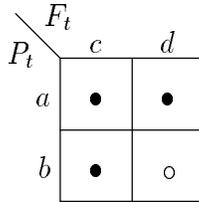


Figure 2: In a rectangular code, the presence of codewords corresponding to a ‘•’ imply the existence of the codeword corresponding to the ‘○’, so that all rectangles in the past/future array can be “filled in” as illustrated.

Example 1 Let C be a group code [2] (i.e., a subgroup of a direct product of symbol groups) regarded as a block code of length two, in which the coordinate p (resp., f) of a codeword (p, f) represents the entire past (resp., future) of that codeword. Suppose $c_1 = (a, d)$, $c_2 = (a, c)$, and $c_3 = (b, c)$ are codewords. As C is a group, the combination $c_1 c_2^{-1} c_3 = (a, d)(a^{-1}, c^{-1})(b, c) = (b, d)$ must be a codeword. Since the split into past and future can be done at an arbitrary depth, this example shows that every group code—and in particular every linear code—is rectangular.

A rectangular code C has the important property that in any reduced trellis presentation for C , if two trellis vertices have a past or future in common, then the vertices are mergeable. More precisely, we have the following theorem.

Theorem 1 *In any reduced trellis T with $C(T)$ rectangular, if v and v' are distinct trellis vertices with $P(v) \cap P(v') \neq \emptyset$ or $F(v) \cap F(v') \neq \emptyset$, then $\{v, v'\}$ is a mergeable pair.*

Proof. Suppose two vertices v and v' have a future z in common, i.e., $z \in F(v) \cap F(v')$. Arbitrarily choose $u \in P(v)$, $w \in P(v')$, $x \in F(v)$, and $y \in F(v')$ as illustrated in Fig. 3. Now since $\{ux, uz, wz\} \subset C(T)$ and $C(T)$ is rectangular, we have $wx \in C(T)$. Since w and x are chosen arbitrarily, we conclude that (a) $P(v')F(v) \subset C$. Similarly, since

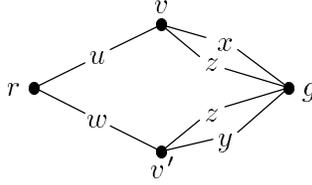


Figure 3: A portion of a trellis.

$\{wy, wz, uz\} \subset C(T)$ we have $uy \in C(T)$; hence we conclude that (b) $P(v)F(v') \subset C$. Together, (a) and (b) imply that $\{v, v'\}$ is a mergeable pair. The case where v and v' have a past in common can be proved similarly, or by applying the same argument to T^R and then reversing trellis edges. ■

B Biproper Trellises

A trellis T is said to be *proper* if the directed edges incident from any trellis vertex are labeled distinctly. In other words, T is proper if, for all distinct edges e_1 and e_2 of T with $i(e_1) = i(e_2)$, we have $\ell(e_1) \neq \ell(e_2)$. A trellis is said to be *co-proper* if the reverse trellis T^R is proper. A trellis that is both proper and co-proper is said to be *biproper*. Of course, if T is biproper, so is T^R . If T is proper, co-proper, or biproper, then any sub-trellis of T is proper, co-proper, or biproper, respectively.

A proper trellis T of length n with root vertex r is “deterministic” in the sense that, for every $d \in [0, n]$ and for every codeword past $s \in P_d[C(T)]$, there is precisely one path p in T from r to a vertex at depth d with $\ell(p) = s$. Indeed, if $s = a_1a_2 \cdots a_d$, then in T there is precisely one edge incident from r to a vertex v_1 with label a_1 , precisely one edge incident from v_1 to a vertex v_2 with label a_2 , etc., and this collection of edges uniquely determines p . We denote by $v(s)$ the unique vertex at depth d in the proper trellis T reached by p .

If T is a co-proper trellis, then T^R is proper and hence deterministic. Applying the argument of the previous paragraph to T^R and then reversing edges shows that in a co-proper trellis T of length n with goal vertex g , for every $d \in [0, n]$ and for every codeword future $s \in F_d[C(T)]$, there is precisely one path p in T from a vertex at depth d to g with $\ell(p) = s$. We denote by $v(s)$ the unique vertex at depth d in the co-proper trellis T from which p starts. We will sometimes say that a co-proper trellis is “co-deterministic.”

We use the fact that a biproper trellis is simultaneously deterministic and co-deterministic to prove the following theorem.

Theorem 2 *A reduced biproper trellis is minimal.*

Proof. Let $T = T(r, g)$ be a reduced biproper trellis, and suppose T contains a mergeable pair $\{v, v'\}$. Let p be a path from r to v and let f be a path from v' to g . Since T is deterministic, there is exactly one path from r with label string $\ell(p)$, and this path must end at v . Similarly, since T is co-deterministic, there is exactly one path ending at g with label string $\ell(f)$, and this path must start at v' . Since $v \neq v'$, it follows that $\ell(p)\ell(f) \notin C(T)$, but this contradicts the mergeability of the pair $\{v, v'\}$. ■

The term “proper trellis” is due to Muder [10]. A proper trellis is “deterministic,” “instantaneously invertible,” “right-resolving,” or “unifilar”—see the multilingual dictionary [13]. Just as every finite state automaton is equivalent to a unique minimal deterministic finite state automaton (Myhill-Nerode theorem) every block code has a unique¹ minimal proper trellis presentation [10]. Applying this result to the reverse code and then reversing trellis edges shows that every block code also has a unique minimal co-proper trellis presentation. The minimal proper trellis presentation is sometimes called the “past-induced canonical realization” [3, 11, 13]. Similarly, the unique minimal co-proper trellis presentation is sometimes called the “future-induced canonical realization.”

In general, the minimal proper and co-proper trellis presentations are distinct, as shown schematically in the poset diagram of Fig. 4(a). (In Fig. 4, the vertices represent trellis presentations for the same code C ; an edge connects a vertex T to a vertex T' below T if and only if T' is isomorphic to a trellis that can be obtained from T by merging two trellis vertices.) However, when the minimal proper and co-proper trellis presentations coincide (forming a biproper trellis), the remarkable situation illustrated in Fig. 4(b) arises, in which the biproper trellis is the *unique* minimal trellis presentation for the given code, which must be rectangular. This situation is stated precisely in the following theorem.

Theorem 3 *For a block code C , the following are equivalent.*

¹Throughout this paper, uniqueness means uniqueness up to isomorphism.

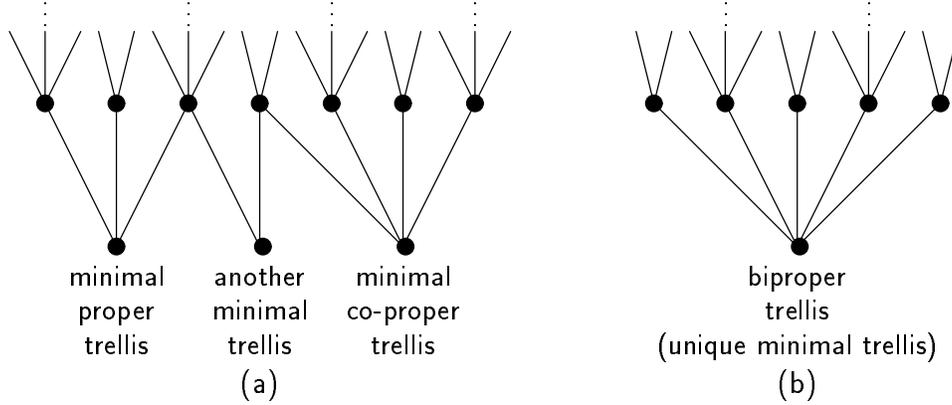


Figure 4: Schematic poset diagram on trellises induced by the partial order relation \preceq (a) in the general case, (b) in the case that a biproper trellis exists.

1. C has a reduced biproper trellis presentation;
2. C is rectangular;
3. C has a unique minimal trellis presentation.

Proof. Suppose C has a reduced biproper trellis presentation $T(r, g)$ of length $n > 1$ and suppose for fixed $t \in [1, n - 1]$ that $\{ac, ad, bc\} \subset C$ for $a, b \in P_t(C)$ and $c, d \in F_t(C)$. Since T is biproper, codeword pasts a and b and codeword futures c and d determine unique vertices $v(a)$, $v(b)$, $v(c)$, and $v(d)$ at depth t in T . However, since $ac \in C$ we have $v(a) = v(c)$. Similarly, since $ad \in C$, $v(a) = v(d)$, and since $bc \in C$, $v(b) = v(c)$. We conclude that $v(b) = v(d)$, hence $bd \in C$, so C is rectangular. Thus (1) implies (2).

Now suppose C is rectangular. If T is a minimal trellis for C that is not co-proper, then there are two edges (v, a, v'') and (v', a, v'') with the same label incident on the same vertex v'' in T . Let z be an arbitrary element of $F(v'')$. Then v and v' have the future az in common, and so $\{v, v'\}$ is a mergeable pair by Theorem 1, contradicting the minimality of T . Hence every minimal trellis for C must be co-proper. However, the minimal co-proper trellis presentation for C is unique. Hence C has a unique minimal trellis, i.e., (2) implies (3).

Suppose C has a unique minimal trellis presentation T . Since T is unique, T is both the minimal proper trellis and the minimal co-proper trellis; hence T is biproper. Thus (3)

implies (1). ■

Since the biproper trellis presentation (T^* , say) is the unique minimal trellis presentation for a rectangular code C , this trellis presentation can be obtained by a sequence of vertex merging operations starting from *any* trellis presentation T for C . In other words, if T is *any* trellis with $C(T) = C(T^*)$ then $T^* \preceq T$, so T^* is a *least* element in the set of trellises partially ordered by \preceq , as shown schematically in Fig. 4(b).

In the terminology of Willems [3], a biproper trellis is a realization that is both past-induced and future-induced. From the equivalent conditions following Theorem 2.2 of [3], it follows that this realization is the unique minimal realization. (Willems' various equivalent conditions—all of which are equivalent to the conditions of Theorem 3—are nicely summarized by Forney and Trott [2, p. 1500].)

C Minimality of Trellis Complexity

The following theorem shows that, among all trellis presentations for a rectangular code, the biproper trellis presentation minimizes a wide variety of structural complexity measures.

Theorem 4 *Let T^* be a reduced biproper trellis of length n , and let T be any trellis with $C(T) = C(T^*)$. Then*

$$\begin{aligned}
 (a) \quad & \forall t \in [0, n], |V_t^*| \leq |V_t|; & (d) \quad & \forall d \in [1, n], |E_d^*| \leq |E_d|; \\
 (b) \quad & |V^*| \leq |V|; & (e) \quad & |E^*| \leq |E|; \\
 (c) \quad & \max_t |V_t^*| \leq \max_t |V_t|; & (f) \quad & \max_d |E_d^*| \leq \max_d |E_d|; \\
 & & (g) \quad & |E^*| - |V^*| \leq |E| - |V|.
 \end{aligned}$$

If any one of inequalities (a), (b), (d), or (e) is achieved with equality, then $T \cong T^$ and all the inequalities (a)–(g) are achieved with equality.*

Proof. See [14] (this issue).

McEliece [9] has shown that the Viterbi algorithm can be used to compute “flows” on a reduced trellis $T = (V, E, A)$, and that the computation requires $|E|$ “multiplications” and $|E| - |V| + 1$ “additions” in a certain semiring. For maximum-likelihood decoding in the Gaussian channel, these semiring “multiplications” and “additions” correspond to real number additions and comparisons, respectively. If we define the total Viterbi decoding

complexity as $2|E| - |V| + 1$, Theorem 4 implies that a biproper trellis T^* has smaller Viterbi decoding complexity than any other trellis presentation for the same code, with equality achieved only by a trellis isomorphic to T^* . Thus, in terms of Viterbi decoding complexity as defined, the biproper trellis is the “best” trellis to use for decoding.

IV Maximal Fixed-Cost Codes

The results of the previous section show that every rectangular code is distinguished by the fact that it admits a unique minimal (biproper) trellis presentation, that not only is minimal with respect to vertex mergeability, but also is minimal with respect to a wide variety of structural complexity measures, including Viterbi decoding complexity. Fortunately, as seen in Example 1, most block codes used in practice, including all linear codes, are rectangular.

In this section we introduce the concept of a “cost framework,” which is used to define the class of maximal fixed-cost (MFC) codes. Our main result is that the MFC codes are rectangular, and include many interesting and useful “nonlinear” codes. Furthermore, we show that MFC codes “capture” the class of rectangular codes in the sense that every rectangular code can be interpreted as an MFC code with respect to some cost framework.

To illustrate the main points, we will keep a simple running example: the familiar case of a linear block code. As we shall see, trellis construction for MFC codes is a generalization of the BCJR/Wolf [5, 6] construction for linear block codes.

A Cost Frameworks

We begin by defining a time-varying “cost function” on the symbol alphabet over which the code is defined. The notion of “cost” here is quite general: symbol costs are required only to be elements of a right-cancellative semigroup.

Recall that a semigroup (S, \circ) is a set S together with an associative binary operation $\circ : S \times S \rightarrow S$. The semigroup (S, \circ) is said to be *right-cancellative* if $a \circ x = b \circ x$ implies $a = b$ for all $a, b, x \in S$. We usually denote a semigroup (S, \circ) simply by S .

Let Π be a product $A_1 \times A_2 \times \cdots \times A_n$ of symbol alphabets. We view A_i as the symbol alphabet at position i . We usually take $A_1 = A_2 = \cdots = A_n$, but this is not required.

Let S be a right-cancellative semigroup with binary operation \circ . Let $\mu = (\mu_1, \dots, \mu_n)$ be a collection of “cost functions” $\mu_i : A_i \rightarrow S$ that associate an element of S with each symbol $a_i \in A_i$. Even if $A_1 = A_2 = \cdots = A_n$, it is not necessary that $\mu_i(a) = \mu_j(a)$ for $i \neq j$. For this reason, we think of the cost function as being “time-varying.”

Example 2 Let $A_1 = A_2 = \cdots = A_n = \mathbb{F}_2$, where \mathbb{F}_2 denotes the finite field with two elements. Then Π is the set of binary n -tuples. Let $H = [h_1, h_2, \dots, h_n]$ be the $r \times n$ parity-check matrix for a binary linear code of length n . We take S to be the set of binary (column) vectors of length r , under binary vector addition. For $a_i \in A_i$, we define $\mu_i(a_i) = a_i \cdot h_i$.

Define the “cost” $\mu(a_1, \dots, a_n)$ of an n -tuple in Π as the product (in S) of coordinate costs, i.e.,

$$\mu(a_1, a_2, \dots, a_n) = \mu_1(a_1) \circ \mu_2(a_2) \circ \cdots \circ \mu_n(a_n),$$

where $a_i \in A_i$ for $i \in [1, n]$. Similarly, for a fixed partition at depth d of an element of Π into past $p = (a_1, \dots, a_d)$ and future $f = (a_{d+1}, \dots, a_n)$, define $\mu(p) = \mu_1(a_1) \circ \cdots \circ \mu_d(a_d)$ and $\mu(f) = \mu_{d+1}(a_{d+1}) \circ \cdots \circ \mu_n(a_n)$.

We call the triple $\Xi = (\Pi, \mu, S)$ that collects together these elements a *cost framework*.

Example 2 (continued). The “cost” associated with a binary n -tuple $\mathbf{a} = (a_1, \dots, a_n)$ is

$$\mu(\mathbf{a}) = \sum_{i=1}^n a_i \cdot h_i = H\mathbf{a}^T,$$

namely, the *syndrome* associated with \mathbf{a} given by the parity-check matrix H .

B Maximal Fixed Cost Codes

Definition. Let $\Xi = (\Pi, \mu, S)$ be a cost framework. For a fixed cost $s \in S$, define the *maximal fixed-cost code* $M_\Xi(s) = \{\mathbf{a} \in \Pi : \mu(\mathbf{a}) = s\}$ to be the set of all possible elements of Π achieving cost s .

Note that every element of $M_{\Xi}(s)$ achieves the same cost s (hence $M_{\Xi}(s)$ is “fixed-cost”) and every element of Π achieving that cost is included (hence $M_{\Xi}(s)$ is “maximal”).

Example 2 (continued). Let $\mathbf{0}$ denote the zero column vector of length r . Then, with Ξ as defined in this running example, $M_{\Xi}(\mathbf{0})$ is the binary linear block code C with parity-check matrix H . For a nonzero r -tuple \mathbf{s} , $M_{\Xi}(\mathbf{s})$ is the coset of C with syndrome \mathbf{s} .

The main motivation for introducing the class of MFC codes is the following theorem, which follows trivially from the right-cancellation property of S , and the definition of MFC codes.

Theorem 5 *Every MFC code $M_{\Xi}(s)$ is rectangular.*

Proof. For partition at depth t of codewords into past and future, let (a, d) , (a, c) , and (b, c) be codewords in $M_{\Xi}(s)$, where $a, b \in P_t(M_{\Xi}(s))$ and $c, d \in F_t(M_{\Xi}(s))$. Then $\mu(a)\mu(c) = \mu(b)\mu(c)$. By right-cancellation, $\mu(a) = \mu(b)$; therefore, $\mu(a)\mu(d) = \mu(b)\mu(d) = s$. Since $\mu(b, d) = s$, and $M_{\Xi}(s)$ is maximal, (b, d) is a codeword, and hence $M_{\Xi}(s)$ is rectangular. ■

The converse is also true, i.e., every rectangular code is an MFC code. More precisely, we have the following theorem, which is proved in Appendix A.

Theorem 6 *For every rectangular code C there exists a cost framework $\Xi = (\Pi, \mu, S)$ with right-cancellative semigroup S and an element $s \in S$, so that $C = M_{\Xi}(s)$.*

When dealing with a given cost framework Ξ , we will usually suppress the subscript ‘ Ξ ’ and write $M(s)$ for the MFC code having cost s .

C Examples

We now show that a wide variety of interesting and useful codes are examples of maximal fixed-cost codes; hence they are rectangular and admit a unique minimal (biproper) trellis presentation.

Example 2 (continued.) (*Linear Codes*) Of course, our running example can be extended to all linear codes. Let $A_i = \mathbb{F}_q$, let $H = [h_1, \dots, h_n]$ be an $r \times n$ matrix with columns h_i having entries from \mathbb{F}_q , and let $S = \mathbb{F}_q^r$ be the vector space of r -tuples over \mathbb{F}_q . For $i \in [1, n]$, define $\mu_i(x) = x \cdot h_i$. Then $\mu(\mathbf{a}) = H\mathbf{a}^T$ is the syndrome associated with \mathbf{a} , and $M(\mathbf{0})$, the set of n -tuples achieving zero syndrome, is the linear code with parity-check matrix H .

Example 3 (*Fixed-Weight Words*) Let $A_i = \{0, 1\}$, let $S = \mathbb{N}_0$ be the non-negative integers under addition, and define μ_i by $\mu_i(0) = 0$; $\mu_i(1) = 1$. Then, for any block length n , $M(w)$ is the set of binary n -tuples of Hamming weight w . This example is a special case of the next one.

Example 4 (*Permutation Codes*) For all $i \geq 1$, let $A_i = \{a_1, \dots, a_c\}$, and let $S = \mathbb{N}_0^c$, the direct product of c copies of \mathbb{N}_0 . Define μ_i by $\mu_i(a_i) = e_i = (0, \dots, 0, 1, 0, \dots, 0)$ where e_i is the unit vector nonzero only in coordinate i . Then $M(m_1, \dots, m_c)$ is the (Variant I) permutation code [15] obtained by permuting the coordinates of the vector of “shape” $(a_1^{m_1}, \dots, a_c^{m_c})$ in all possible ways, where $a_i^{m_i}$ denotes the m_i -tuple (a_i, a_i, \dots, a_i) .

If the alphabet is extended so that $A_i = \{\pm a_1, \dots, \pm a_c\}$, and $\mu_i(\pm a_i) = e_i$, then $M(m_1, \dots, m_c)$ is the Variant II permutation code, in which arbitrary coordinate sign changes are permitted, in addition to coordinate permutations.

Example 5 (*Lattice Shells*) Let $A_i = \mathbb{Z}$, let $S = \mathbb{N}_0$, and, for all $i \geq 1$, define μ_i by $\mu_i(x) = x^2$. Then, for any block length n , $M(w)$ is the set of integer n -tuples of squared norm w , i.e., a shell in the integer lattice \mathbb{Z}^n .

More generally, for all $i \geq 1$, let $\mu_i : \mathbb{Z} \rightarrow \mathbb{R}$ be any real “per-letter cost” function. Then, for any block length n , $M(w)$ is the integer lattice shell attaining total cost w . In particular, choosing $\mu_i(x) = |x|^p$ shows that, for any p , the integer lattice shell with respect to the L_p norm is rectangular.

D Product Cost Frameworks

Let $\Xi(\Pi, \mu, S)$ and $\Xi'(\Pi, \mu', S')$ be two cost frameworks with the same alphabet set $\Pi = A_1 \times \cdots \times A_n$, with $\mu = (\mu_1, \dots, \mu_n)$ and $\mu' = (\mu'_1, \dots, \mu'_n)$. The *product* cost framework $\Xi \times \Xi'$ is defined as $\Xi \times \Xi' = (\Pi, \mu \times \mu', S \times S')$ where $\mu \times \mu' = (\mu_1 \times \mu'_1, \dots, \mu_n \times \mu'_n)$ with $\mu_i \times \mu'_i : A_i \rightarrow S \times S'$ defined by $a \mapsto (\mu_i(a), \mu'_i(a))$.

If $M_{\Xi}(s)$ and $M_{\Xi'}(s')$ are MFC codes with respect to cost frameworks Ξ and Ξ' , respectively, then it is easy to see that

$$M_{\Xi}(s) \cap M_{\Xi'}(s') = M_{\Xi \times \Xi'}(s, s').$$

In other words, the intersection of two MFC codes with respect to two different cost frameworks over the same alphabet is an MFC code with respect to the product of the cost frameworks.

Example 6 (*Fixed Weight Subcodes*) Combining Example 2 with Example 3 shows that a fixed weight subcode of a linear code is an MFC code, and hence is rectangular.

V Trellis Construction for MFC Codes

In this section, we show that the minimal trellis for an MFC code can be constructed as a subtrellis of an unreduced trellis called the “complete cost trellis” associated with the given cost framework.

A Complete Cost Trellises

Let $\Xi = (\Pi, \mu, S)$ be a cost framework, with $\Pi = A_1 \times \cdots \times A_n$, and $\mu = (\mu_1, \dots, \mu_n)$. If S does not contain an identity element, adjoin an element ‘1’ to S with the property that $1s = s1 = s$ for all $s \in S$, and let S^1 denote the monoid (i.e., semigroup with identity) so obtained. If S contains an identity, denote the identity by ‘1’ and let $S^1 = S$.

The *complete cost trellis* for the cost framework Ξ is constructed as follows. Let $S_0 = \{1\}$, and, for $d \in [1, n]$, let

$$S_d = \{s \in S : \mu(a_1, a_2, \dots, a_d) = s \text{ for some } (a_1, a_2, \dots, a_d) \in P_d(\Pi)\}$$

be the set of all possible “intermediate” costs achievable by strings of length d from $P_d(\Pi)$.

At each depth $d \in [0, n]$, we wish to associate a distinct trellis vertex with each distinct element of S_d . To accomplish this, we define the map $v_d : S_d \rightarrow S_d \times [0, n]$ by $v_d(s) = (s, d)$. Observe that v_d simply “tags” each cost with the corresponding depth d . We take V_d , the set of vertices at depth d , as $V_d = v_d(S_d)$, the image of S_d under this map, i.e., as the set of costs with “tag” d . Overloading the symbol μ still further, for $d \in [0, n]$, we define the map $\mu : V_d \rightarrow S$ by $\mu(s, d) = s$; then for a vertex $v \in V_d$, $\mu(v)$ denotes the intermediate cost associated with v .

For $d \in [1, n]$, let

$$E_d = \{(v, a, v') \in V_{d-1} \times A_d \times V_d : \mu(v)\mu_d(a) = \mu(v')\}.$$

This set of edges connects a vertex v at depth $d - 1$ having cost $\mu(v)$ to a vertex v' at depth d having cost $\mu(v')$ by an edge with label a if and only if $\mu(v)\mu_d(a) = \mu(v')$.

Let $V = \cup_{d=0}^n V_d$, $E = \cup_{d=1}^n E_d$, and $A = \cup_{i=d}^n A_d$. The trellis $T = (V, E, A)$ is the complete cost trellis for the cost framework Ξ . Note that the complete cost trellis is a trellis for the partition of all possible sequences of length n into same-cost equivalence classes. Similarly, the trellis up to depth d is a trellis for the partition of the sequences of length d into same-cost equivalence classes.

Example 7 (*Linear Codes*) As in Section C, Example 2, let

$$\begin{aligned} H &= [h_1, \dots, h_7] \\ &= \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \end{aligned}$$

be a parity check matrix for a (7,4) Hamming code. Take $A_i = \mathbb{F}_2$, $S = \mathbb{F}_2^3$, and $\mu_i(x) = x \cdot h_i$. Fig. 5 shows the complete cost trellis for this framework.

Example 8 (*Fixed Weight Words*) As in Section C, Example 3, let $n = 7$, $A_i = \{0, 1\}$, $S = \mathbb{Z}$, and take $\mu_i(0) = 0$, $\mu_i(1) = 1$. Fig. 6 shows the complete cost trellis for this framework.

Example 9 (*Permutation Codes*) As in Section C, Example 4, let $A_i = \{a, b, c\}$, and $S = (\mathbb{N}_0)^3$. For all i , define $\mu_i(a) = (1, 0, 0)$, $\mu_i(b) = (0, 1, 0)$, and $\mu_i(c) = (0, 0, 1)$. Fig. 7 shows the complete cost trellis for this framework for lengths up to $n = 4$. Setting g to a vertex labeled (i, j, k) yields a trellis for the Type I permutation code of length $i + j + k$ with “shape” (a^i, b^j, c^k) .

Example 10 (*Lattice Shells*) As in Section C, Example 5, let $n = 4$, $A_i = \mathbb{Z}$, $S = \mathbb{N}_0$, and $\mu_i(x) = x^2$. Fig. 8 shows a portion of the complete cost trellis for this framework.

Remark. Each vertex v in Fig. 8 is labeled with $|P(v)|$. Reading off these labels from bottom to top at any fixed trellis depth d yields the theta series coefficients [16] for the integer lattice in d dimensions. For example, $\Theta_{\mathbb{Z}^2}(q) = 1 + 4q + 4q^2 + 0q^3 + 4q^4 + \dots$.

Theorem 7 *A complete cost trellis is biproper.*

Proof. By construction, T is proper, because for any $v \in V_{i-1}$, and any $a \in A_i$, the only edge labeled a incident from v is incident to $v_i(\mu(v)\mu_i(a))$. In fact T is also co-proper, since if (v', a, v) and (v'', a, v) are edges incident on $v \in V_i$, we have $\mu(v) = \mu(v')\mu_i(a) = \mu(v'')\mu_i(a)$; hence by right-cancellation in S , $\mu(v') = \mu(v'')$, and since v' and v'' are at the same depth in T , we must have $v' = v''$. ■

B Extracting the Minimal Trellis for an MFC Code

Every subtrellis of a biproper trellis is biproper. Let $r = (1, 0)$ be the unique vertex at depth zero in the complete cost trellis for cost framework Ξ , and let g be any depth n vertex with $\mu(g) = s$. Then the reduced trellis $T(r, g)$ is biproper, and hence the minimal trellis for $M_\Xi(s)$. In other words, the minimal trellis for an MFC code defined with respect to a given cost framework can be obtained from the complete cost trellis simply by deleting all vertices that are not essential with respect (r, g) .

For example, the solid edges of Fig. 5 constitute the minimal trellis for the (7,4) Hamming code. This method of constructing trellises for linear block codes was also used by Bahl, *al.* [5] and by Wolf [6]. Another example is given in Fig. 6, in which the solid edges

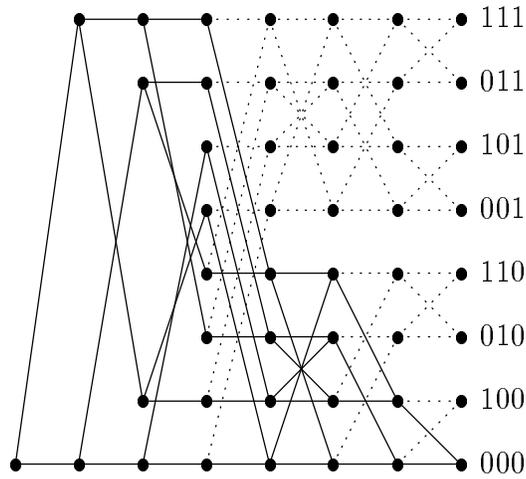


Figure 5: The complete cost trellis for a $(7,4)$ Hamming code. Horizontal edges have label '0'; diagonal edges have label '1'. Vertices at the same horizontal level have the same intermediate cost, as indicated. Dotted edges are those not included in any path from r to the vertex at depth 7 with cost 000.

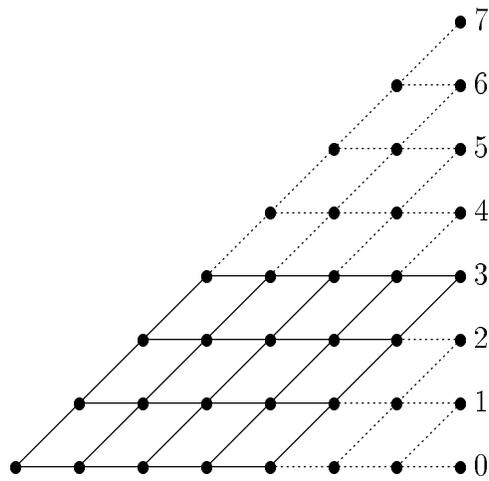


Figure 6: The complete cost trellis for binary 7-tuples with Hamming weight symbol cost. Horizontal edges have label '0'; diagonal edges have label '1'. Vertices at the same horizontal level have the same intermediate cost, as indicated. Dotted edges are those not included in any path from r to the vertex at depth 7 with cost 3.

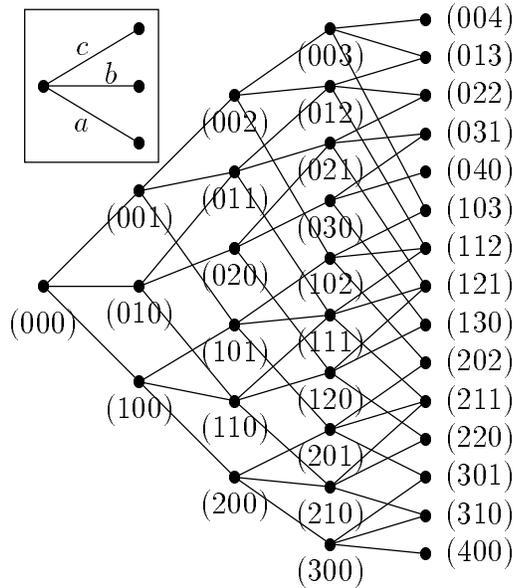


Figure 7: The complete cost trellis (up to length 4) for permutation codes over the alphabet $\{a, b, c\}$, with intermediate costs labeled. Edges are labeled as shown in the inset.

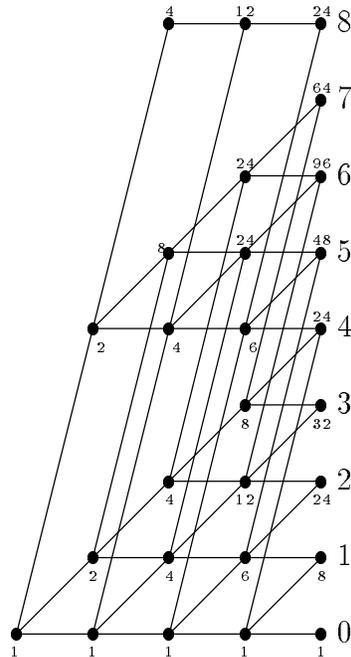


Figure 8: A portion of the complete cost trellis for integer four-tuples with $\mu_i(x) = x^2$. Horizontal edges have label '0'; diagonal edges of slope 1 and 4 represent parallel edges with labels ± 1 and ± 2 , respectively.

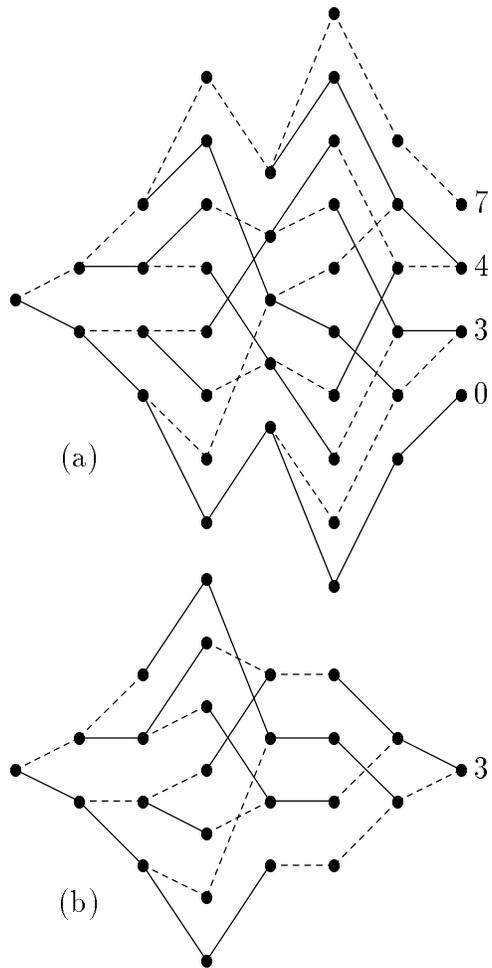


Figure 9: (a) Intersection of Hamming code minimal trellis with complete fixed weight trellis. (b) Reduced biproper trellis for Hamming codewords of weight 3. Solid edges are labeled '0', dashed edges are labeled '1'.

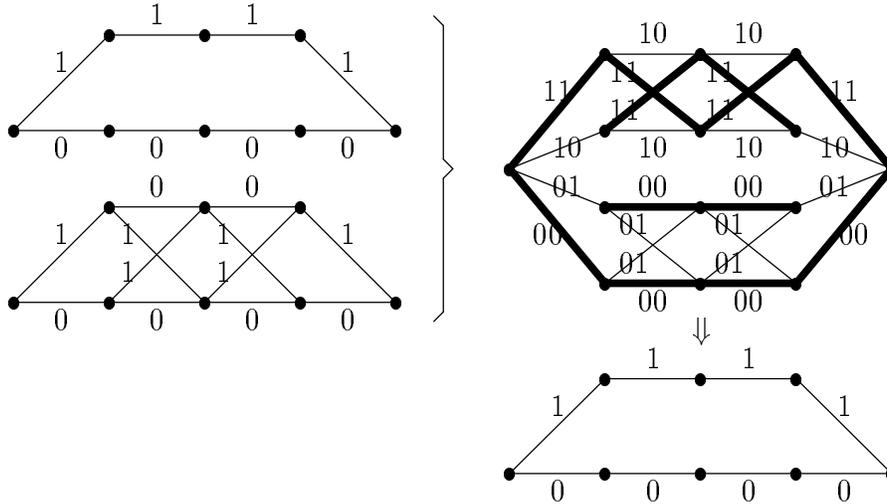


Figure 10: Obtaining a trellis description for the intersection of two codes.

constitute the minimal trellis for binary 7-tuples of Hamming weight 3. Minimal trellises for other MFC codes can be extracted from Figs. 6–8.

Example 11 (*A Product Cost Framework*) As a final example, Fig. 9(a) shows a subtrellis of the complete cost trellis associated with the product of the cost frameworks of Examples 7 and 8. Shown are the trellis edges contributing to paths that achieve zero syndrome, and Hamming weight w . The words of minimum Hamming weight form an MFC code, the minimal trellis of which is shown in Fig. 9(b).

The trellis corresponding to the intersection of MFC codes as in Example 11 can also be constructed using a generalization of the trellis product (sometimes called the Shannon product) defined in [1, 17]. Instead of *adding* edge labels in the product trellis as in the trellis construction for linear codes [1], simply map the product of two edges as follows: $(v, a, v') \times (w, b, w') \mapsto (vw, ab, v'w')$. The resulting trellis describes the code obtained by interleaving the codes corresponding to the trellises being multiplied. The trellis describing the intersection of the two codes can be obtained first by deleting all edges that are not labelled with a repeated symbol, and then by deleting all remaining edges that are not included in a path from r to g . The repeated label xx is then mapped back to x . Fig. 10 illustrates this procedure for two length-four binary linear codes.

VI Conclusions

The main result of this paper is Theorem 5 which states that every maximal fixed-cost code is rectangular, and hence admits a biproper trellis presentation. The converse is also true: Theorem 6 states that, corresponding to every rectangular code C , there is a cost framework such that C is an MFC code with respect to this cost framework.

Many examples of useful objects in coding theory, like linear codes, subcodes of constant weight, permutation codes, shells of constant weight in an integer lattice, etc., are MFC codes, and hence are rectangular. These codes have unique minimal trellis presentations, and the minimal trellis minimizes a variety of trellis complexity measures, including edge count, vertex count, and Viterbi decoding complexity. No doubt, by choosing an appropriate cost framework, many other examples of interesting and useful codes can be shown to be rectangular.

Trellises for maximal fixed-cost codes are useful for the insight they give into the structure of the codes that they describe. The trellises may also be useful for such tasks such as error detection (testing a sequence for membership in a code), decoding, sequence detection, and encoding. Schalkwijk's indexing scheme for permutation codes [18] and other indexing schemes known in the literature on combinatorial algorithms [19, Ch. 13] can be viewed as being trellis-based. Such schemes are used to compute bijections between integers and trellis paths and hence codewords. The "shell-mapping" algorithm [20, 21, 22, 23] incorporated in the V.34 modem standard can also be viewed as a trellis-based indexing scheme.

An interesting avenue for possible future work in this area is to extend the cost framework approach to sequence codes. This could be done, for example, by admitting sequences achieving a range of intermediate costs at any particular trellis depth. Many interesting sequence codes (for example, codes with a bounded running digital sum) would fit naturally into such a framework.

Appendix A. A Converse Theorem

We prove that for every rectangular code C of length n there exists a cost framework (Π, μ, S) with right-cancellative semigroup S , so that $C = M(s)$ is an MFC code for some fixed element $s \in S$.

Let $T = (V, E, A)$ be the reduced biproper trellis presentation for C , and, for $d \in [1, n]$, let $L_d = \{\ell(e) : d(f(e)) = d, e \in E\}$ be the set of labels that occur on edges incident to vertices with depth d . Let $N = [1, n]$. Define the map $w_d : L_d \rightarrow L_d \times N$ by $w_d(a) = (a, d)$. Let $w_d(L_d) = \{w_d(a) : a \in L_d\}$. Let $D = \bigcup_{d=1}^n w_d(L_d)$; effectively, D is the *disjoint union* of the L_d . Define the projection map $\lambda : D \rightarrow A$ by $\lambda(a, d) = a$ and the projection map $\delta : D \rightarrow N$ by $\delta(a, d) = d$. These maps return, respectively, the *label* and *depth in T* of an element of D .

Let D^* be the free monoid generated by D , i.e., the set of finite length strings with symbols from D , closed under concatenation of strings. The identity in D^* is the empty string ϵ .

Elements of D^* are called *words*. In D^* , if $w = xy$, then x is said to be a *prefix* of w and y is said to be a *suffix* of w . The length of (i.e., number of symbols of D contained in) a word w is denoted by $|w|$. We have $|\epsilon| = 0$, and for all $x, y \in D^*$, $|xy| = |x| + |y|$. A word $x = x_1x_2 \cdots x_m$ of length m is said to have *subword* $x[i, j] = x_ix_{i+1} \cdots x_j$ if $1 \leq i \leq j \leq m$. Two subwords $x[a, b]$ and $x[c, d]$ of a word x are *disjoint* if $b < c$ or $d < a$.

A word $w = a_1a_2 \cdots a_k$ of length k has *label string* $\lambda(w) = \lambda(a_1)\lambda(a_2) \cdots \lambda(a_k)$ and *depth sequence* $\delta(w) = (\delta(a_1), \delta(a_2), \dots, \delta(a_k))$. We have $\lambda(xy) = \lambda(x)\lambda(y)$ and $\delta(xy) = (\delta(x), \delta(y))$. We say that a word w of length $d \leq n$ is a “codeword past” (and write, by abuse of notation, that $w \in P_d(C)$) if $\lambda(w) \in P_d(C)$ and $\delta(w) = (1, 2, \dots, d)$. Recall from Section III.B that since T is proper, a codeword past $p \in P_d(C)$ determines a path from r to a unique vertex $v(p)$ at depth d in T . Similarly, for a word w such that $w \in P_d(C)$, define $v(w) = v(\lambda(w))$.

A *replaceable subword* (RS) of a word is a subword \tilde{x} of length d such that $\tilde{x} \in P_d(C)$. A *maximal RS* of a word x is an RS of maximal length, i.e., an RS $\tilde{x} = x[i, j]$ with $j = |x|$ or $x[i, j + 1]$ not an RS. If \tilde{x}_1 and \tilde{x}_2 are distinct maximal replaceable subwords of x , then

\tilde{x}_1 and \tilde{x}_2 are disjoint.

We define a relation on words as follows. For words $\alpha = \alpha_1\alpha_2\alpha_3$ and $\beta = \alpha_1\beta_2\alpha_3$, write $\alpha R\beta$ (α “is replaceable by” β) if there exists $w \in D^*$ such that $\alpha_2w \in C$ and $\beta_2w \in C$. In other words, $\alpha R\beta$ if α_2 and β_2 are codeword pasts with $v(\alpha_2) = v(\beta_2)$. (Necessarily $|\alpha_2| = |\beta_2|$, so $|\alpha| = |\beta|$.)

If $\alpha = \beta$ then $\alpha R\beta$ (set $\alpha_1 = \alpha$, and $\alpha_2 = \alpha_3 = \epsilon$ in the previous paragraph) so R is reflexive. Furthermore, R is symmetric, i.e., if $\alpha R\beta$ then $\beta R\alpha$. If $\alpha R\beta$ then $\delta(\alpha) = \delta(\beta)$.

Lemma 1 *If $x[i, j]$ is a maximal RS of x , and xRy , then $y[i, j]$ is a maximal RS of y , and $v(x[i, j]) = v(y[i, j])$.*

Proof. Suppose xRy , then $x = \alpha_1\alpha_2\alpha_3$, $y = \alpha_1\beta_2\alpha_3$, with α_2 and β_2 replaceable subwords and $v(\alpha_2) = v(\beta_2)$. Let $a = |\alpha_1| + 1$ and $b = |x| - |\alpha_3|$. Then $\alpha_2 = x[a, b]$.

If $j < a$ or $i > b$, $x[i, j] = y[i, j]$, so $y[i, j]$ is an RS of y , with $v(x[i, j]) = v(y[i, j])$. Otherwise, we must have $a = i$, so $x[i, j] = \alpha_2w$ and $y[i, j] = \beta_2w$ for some w . Since $v(\alpha_2) = v(\beta_2)$, we have, in any proper trellis, $v(\alpha_2w) = v(\beta_2w)$, as illustrated in Fig. 11. Therefore, since $x[i, j]$ is an RS of x , $y[i, j]$ is an RS of y .

Now, if $j = |y|$, then $y[i, j]$ is maximal. If $j < |y|$, let $e = y[j + 1, j + 1]$, and note that $x[j + 1, j + 1] = e$ also. If $y[i, j]e$ is an RS, then so must be $x[i, j]e$, contradicting the maximality of $x[i, j]$. So $y[i, j]e$ is not an RS and hence $y[i, j]$ is maximal. \blacksquare

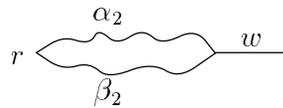


Figure 11: A subgraph of T .

The proof of Lemma 1 shows that a replacement ‘involves’ exactly one maximal RS of a word, with all other maximal RS’s left unchanged. More precisely, suppose xRy , and let $RS(x) = \{[i_1, j_1], \dots, [i_m, j_m]\}$ denote the set of *intervals* corresponding to maximal replaceable subwords of a word x . The proof of Lemma 1 implies that for at most one value of k is $x[i_k, j_k] \neq y[i_k, j_k]$. Furthermore, denote by $\overline{RS}(x) = \{k \in [1, |x|], k \notin \cup RS(x)\}$ the complement of $RS(x)$ relative to $[1, |x|]$. Then for all $k \in \overline{RS}(x)$, $x[k, k] = y[k, k]$, i.e., substrings that are not part of a maximal RS are not changed by a replacement.

For words α and β , we will write $\alpha =_R \beta$ if there exists a finite sequence (called a *deduction*)

$$\alpha R \alpha_1, \alpha_1 R \alpha_2, \dots, \alpha_{k-1} R \beta$$

of replacements which “transform” α to β . If $\alpha =_R \beta$, then $|\alpha| = |\beta|$ and $\delta(\alpha) = \delta(\beta)$. It is easy to verify that the relation ‘ $=_R$ ’ is an equivalence relation on D^* . In fact, $=_R$ is a *congruence* on D^* , since, for all $x, y, z \in D^*$ with $x =_R y$, we have $xz =_R yz$ and $zx =_R zy$.

The discussion following the proof of Lemma 1 implies the following.

Lemma 2 $x =_R y$ if and only if, for all intervals $[i, j] \in RS(x)$, $x[i, j] R y[i, j]$, and for all $k \in \overline{RS}(x)$, $x[k, k] = y[k, k]$.

Define $\rho(x) = \{y \in D^* : x =_R y\}$ to be the equivalence class containing x . Let $D^*/(=_R)$ denote the set of equivalence classes. For $X, Y \in D^*/(=_R)$, take $x \in X$ and $y \in Y$ and define $X \cdot Y = \rho(xy)$. This operation is well defined, since if $x_1 =_R x$ and $y_1 =_R y$ then, using the fact that $=_R$ is a congruence, $x_1 y_1 =_R x y$, so all these elements define the same equivalence class. We let $S = (D^*/(=_R), \cdot)$; then S is a semigroup.

Theorem 8 S is right-cancellative.

Proof. It is enough to show that $X \cdot \rho(z) = Y \cdot \rho(z)$ implies $X = Y$, where z is a word of length one. Take $x \in X$ and $y \in Y$; then $xz =_R yz$. We must show that $x =_R y$. By Lemma 2, $(xz)[i, j] R (yz)[i, j]$ for all $[i, j] \in RS(xz)$, and if $k \in \overline{RS}(xz)$, then $(xz)[k, k] = (yz)[k, k]$.

If z is not the suffix of a maximal RS of xz of length > 1 , then $RS(x) \subset RS(xz)$, so $x[i, j] R y[i, j]$ for all $[i, j] \in RS(x)$. For $k < |xz|$, $x[k, k] = y[k, k]$. Therefore, by Lemma 2 $x =_R y$.

Suppose now that z is the suffix of a maximal RS \tilde{x} of xz of length > 1 , so $\tilde{x} = \alpha z$. By Lemma 1, z is the suffix of a maximal RS \tilde{y} of y , so $\tilde{y} = \beta z$, and $v(\alpha z) = v(\beta z)$. But since T is co-proper, this implies $v(\alpha) = v(\beta)$, and therefore, $\alpha R \beta$. All other maximal replaceable subwords of xz are also maximal replaceable subwords of x ; therefore $x =_R y$. ■

We have constructed a right cancellative semigroup S . For any element $a \in A$, let $\mu_i(a) = \rho(a, i)$. Define the map $\omega : C \rightarrow D^*$ by $\omega(c_1 c_2 \cdots c_n) = (c_1, 1)(c_2, 2) \cdots (c_n, n)$ and

let $s = \rho(\omega(c))$ for some codeword $c \in C$.

For $u \in C$, we have $\omega(u) =_R \omega(c)$, so $\rho(\omega(u)) = \rho(\omega(c)) = s$; hence $C \subset M(s)$. On the other hand, for any $u \in M(s)$, we have $\omega(u) =_R \omega(c)$, so $v(u) = v(c)$ and hence $u \in C(T) = C$. Therefore $M(s) \subset C$. We conclude that $M(s) = C$, and that C is an MFC code.

Acknowledgments

I am indebted to Prof. J. McCool of the Department of Mathematics, University of Toronto, for pointing me to the general framework in which the proof of Theorem 6 could be carried out, and to the anonymous reviewers and guest editor G. David Forney, Jr., for helpful comments that led to an improved paper.

References

- [1] F. R. Kschischang and V. Sorokine, "On the trellis structure of block codes," *IEEE Trans. on Inform. Theory*, vol. 41, pp. 1924–1937, Nov. 1995.
- [2] G. D. Forney, Jr. and M. D. Trott, "The dynamics of group codes: State spaces, trellis diagrams and canonical encoders," *IEEE Trans. on Inform. Theory*, vol. 39, pp. 1491–1513, Sept. 1993.
- [3] J. C. Willems, "Models for dynamics," in *Dynamics Reported, Volume 2* (U. Kirchgraber and H. O. Walther, eds.), pp. 171–269, John Wiley and Sons, 1989.
- [4] G. D. Forney, Jr., "Review of random tree codes." Nasa Ames Research Center, Appendix A of Final Report on Contract NAS2-3637, NASA CR73176, Dec. 1967.
- [5] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. on Inform. Theory*, vol. 20, pp. 284–287, Mar. 1974.

- [6] J. K. Wolf, “Efficient maximum-likelihood decoding of linear block codes using a trellis,” *IEEE Trans. on Inform. Theory*, vol. IT-24, pp. 76–80, 1978.
- [7] J. L. Massey, “Foundation and methods of channel encoding,” in *Proc. Int. Conf. Inform. Theory and Systems*, vol. 65, (Berlin), Sept. 1978.
- [8] G. D. Forney, Jr., “Coset codes II: Binary lattices and related codes,” *IEEE Trans. on Inform. Theory*, vol. 34, pp. 1152–1187, Sep. 1988.
- [9] R. J. McEliece, “On the BCJR trellis for linear block codes,” *IEEE Trans. on Inform. Theory*, vol. 42, 1996. To appear.
- [10] D. J. Muder, “Minimal trellises for block codes,” *IEEE Trans. on Inform. Theory*, vol. 34, pp. 1049–1053, Sept. 1988.
- [11] M. D. Trott, *The Algebraic Structure of Trellis Codes*. PhD thesis, Dept. Elec. Engg., Stanford University, Stanford, CA, Aug. 1992.
- [12] H.-A. Loeliger, G. D. Forney, Jr., T. Mittelholzer, and M. D. Trott, “Minimality and observability of group systems,” *Linear Algebra Appl.*, vol. 205–206, pp. 937–963, July 1994.
- [13] G. D. Forney, Jr., B. Marcus, N. T. Sindhushayana, and M. D. Trott, “Multilingual dictionary,” in *Different Aspects of Coding Theory* (A. R. Calderbank, ed.), vol. 50 of *Proc. of Symposia in Applied Mathematics*, pp. 109–138, American Mathematical Society, 1995.
- [14] A. Vardy and F. R. Kschischang, “Proof of a conjecture of McEliece regarding the expansion index of the minimal trellis,” *IEEE Trans. on Inform. Theory*, vol. 42, 1996. To appear.
- [15] D. Slepian, “Permutation modulation,” *Proc. IEEE*, vol. 53, pp. 228–236, Mar. 1965.
- [16] J. H. Conway and N. J. A. Sloane, *Sphere Packings, Lattices and Groups*. New York: Springer-Verlag, 2nd ed., 1993.

- [17] V. Sidorenko, G. Markarian, and B. Honary, “Code trellises and the Shannon product,” in *Proc. Seventh Joint Swedish-Russian Int. Workshop on Inform. Theory*, (St. Petersburg, Russia), pp. 220–224, June 1995.
- [18] J. P. M. Schalkwijk, “An algorithm for source coding,” *IEEE Trans. on Inform. Theory*, vol. IT-18, pp. 395–399, May 1972.
- [19] A. Nijenhuis and H. S. Wilf, *Combinatorial Algorithms*. New York: Academic Press, 2nd ed., 1978.
- [20] G. R. Lang and F. M. Longstaff, “A Leech lattice modem,” *IEEE J. on Selected Areas in Commun.*, vol. 7, pp. 968–973, Aug. 1989.
- [21] F. R. Kschischang and S. Pasupathy, “Optimal shaping properties of the truncated polydisc,” *IEEE Trans. on Inform. Theory*, vol. 40, pp. 892–903, May 1994.
- [22] R. Laroia, N. Farvardin, and S. Tretter, “On optimal shaping of multidimensional constellations,” *IEEE Trans. on Inform. Theory*, vol. 40, pp. 1044–1056, July 1994.
- [23] A. K. Khandani and P. Kabal, “Shaping multidimensional signal spaces—Part I: Optimum shaping, shell mapping,” *IEEE Trans. on Inform. Theory*, vol. 39, pp. 1799–1808, Nov. 1993.