Combinatorial Representations of Linear Block Codes

Priti Shankar

Computer Science and Automation
Indian Institute of Science, Bangalore

October 12, 2007

Collaborator: Aditya Nori
Outline

• Algebraic Descriptions of Linear Block Codes.

• Trellis Structure of Block Codes.

• Tail-Biting Trellises.

• Algebraic Properties of Tail-Biting Trellises
Linear Block Codes

A linear \((n, k)\) block code is a \(k\)-dimensional subspace of an \(n\)-dimensional vector space. Typically defined by a generator matrix which is a \(k \times n\) matrix whose rows form the basis for the subspace. A code generated by this matrix is a \((n, k)\) code. For example, a binary \((7, 4)\) Hamming Code can be described by the following generator matrix

\[
G = \begin{bmatrix}
0 & 0 & 0 & 1 & 1 & 0 & 1 \\
1 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

Here the code space has 16 vectors. The information rate of the code is \(k/n\). Thus, this Hamming code has a rate of 4/7.
The Parity Check Matrix

A linear block code can also be described by a set of vectors that are orthogonal to the code vectors. These are called constraints or parity checks and can be defined by a matrix $H$.

If $(c_1, c_2, \ldots c_7)$ represents a codeword, the parity check matrix specifies a set of three constraints on each codeword.
Constraints implied by the parity check matrix

\[ H = \begin{bmatrix}
  1 & 1 & 0 & 0 & 1 & 0 & 1 \\
  1 & 1 & 1 & 0 & 0 & 1 & 0 \\
  0 & 1 & 1 & 1 & 0 & 0 & 1
\end{bmatrix}. \]

\[ c_1 + c_2 + c_5 + c_7 = 0 \]
\[ c_1 + c_2 + c_3 + c_6 = 0 \]
\[ c_2 + c_3 + c_4 + c_7 = 0 \]

It is easy to see that the matrix \( H \) defines a space that is orthogonal to the code space, or \( GH^T = 0 \). If the rows of \( H \) are linearly independent, then \( H \) is a \((n - k) \times n\) matrix.

The vector space has 128 vectors of which 16 are codewords, each of which satisfies 3 linearly independent parity checks.
Dual Codes

If $G$ and $H$ are the generator and parity check matrices of an $(n, k)$ linear block code then $H$ and $G$ are the generator and parity check matrices respectively of the dual $(n, n - k)$ code.

Thus the dual of a $(7, 4)$ binary Hamming code is a $(7, 3)$ binary linear code.

We denote the dual of a linear block code $C$ by $C^\perp$. 

Trellis Representations of Block Codes

A conventional trellis $T = (V, E, F_2)$ of depth $n$ is an edge-labeled directed graph and:

- The vertex set $V$ can be partitioned into $n+1$ vertex classes
  
  $V = V_0 \cup V_1 \cup \cdots \cup V_n$ where $|V_0| = |V_n| = 1$

- Every edge in $T$ is labeled with a symbol from the alphabet $F_2$, and begins at a vertex of $V_i$ and ends at a vertex of $V_{i+1}$, for some $i \in \{0, 1, \ldots, n-1\}$.

- The length of a path (in edges) from the root to any vertex is unique and the set of indices $\mathcal{I} = \{0, 1, \ldots, n\}$ for the partition above are the time indices.
• $|V_i|$ is the state-cardinality of the trellis at time index $i$ and the sequence 
$\{|V_i|\}, \ 0 \leq i \leq n$ defines the state-cardinality profile (SCP) of the trellis. 
We will denote by $S_{\text{max}}(T)$ the maximum state-cardinality of $T$ over all 
time indices.
Figure 1: A Conventional Trellis for the (7,4) Hamming Code

Time index sequence: (0,1,2,3,4,5,6,7)

State Cardinality Profile: (1,2,4,8,8,4,2,1), $S_{max} = 8$
Definitions

• The trellis $T$ is said to represent a block code $C$ over $F_2$ if the set of all edge-label sequences in $T$ is equal to $C$. Let $C(T)$ denote the code represented by the trellis $T$.

• A trellis $T$ for a code $C$ of length $n$ is minimal if it satisfies the following property: for each $i = 0, 1, \ldots, n$, the number of vertices in $T$ at time index $i$ is less than or equal to the number of vertices at time index $i$ in any other trellis for $C$.

• A trellis is said to be biproper if any pair of edges directed towards a vertex has distinct labels (co-proper), and so also any pair of edges leaving a vertex (proper).

Theorem (Muder 88): Every linear block code has a unique minimal trellis which is biproper.
**Trellis Construction Algorithms**

**Definition.** The *linear span* of vector $c$ is the smallest interval $[i, j]$, $i, j \in \{1, 2, \ldots, n\}$, $i < j$, that contains all the non-zero positions of $c$. For example, $0001101$ has linear span $[4, 7]$.

**Definition.** For a vector $x$ over the field $\mathbb{F}_2$ with span $[a, b]$, there is a unique *elementary trellis* representing $\langle x \rangle$, the vector space generated by $x$ [Kschischang and Sorokine, 1995]. This trellis has 2 vertices at those positions that belong to $[a, b)$, and a single vertex at other positions.

![Elementary trellis for (0110, [2, 3])](image)
The Product Construction

Figure 2: Elementary trellises for rows (0110) and (1001) with spans [2,3] and [1,4] respectively and corresponding product trellis
The Minimal Trellis

How does one construct the minimal trellis from a generator matrix for the code?
The minimal trellis is constructed from a generator matrix by putting it in trellis oriented form.

No two spans begin or end in the same position Two spans begin in the same position iff the product trellis is non-proper. Two spans end in the same position iff the product trellis is non-coproper.

The process is similar to performing two sequences of elementary row operations and requires polynomial time. Can read off the state cardinality profile from the set of spans. For a binary code, every index in 
\[(a, b)\]
for a span 
\([a, b]\)
contributes a factor of 2 to the cardinality at that index.
The BCJR Construction (Bahl, Cocke, Jelinek, Raviv, 1974)

Let \( H = [h_1 h_2 \cdots h_n] \) be an arbitrary parity check matrix for an \((n, k)\) linear block code \( C \) over the binary field where \( h_i \) represents the \( i^{th} \) column of \( H \).

The vertex set \( V_i \) for \( C \) is constructed by identifying vertices in \( V_i \) with partial syndromes taken with respect to the first \( i \) columns of \( H \). Specifically: Every codeword \( c = (c_1, c_2, \ldots, c_n) \in C \) induces a sequence of states \( \{s_i\}_{i=0}^{n-1} \), each state being labeled by a \((n - k) \times 1\) vector as follows.

\[
s_i = \begin{cases} 
0 & \text{if } i = 0 \\
 s_{i-1} + c_i h_i & \text{otherwise}
\end{cases}
\]
There is an edge labeled $a \in \mathbb{F}_2$ from state $s_{i-1}$ to state $s_i$, $1 \leq i \leq n-2$, if and only if

$$s_i = s_{i-1} + ah_i$$

Since the syndrome of each codeword is 0 we have $V_n = \{0\}$
Linearity of State Spaces

Let $H_i$ and $G_i$ be the matrices of the first $i$ columns of $H$ and $G$ respectively.

Then it is obvious from the definition that $V_i$ is the column space of $H_iG_i^T$, an $(n - k) \times k$ matrix. Thus the set of vertex labels at time $i$ is a linear space for all $i$.

If we read off vertex labels and edge labels as we traverse a codeword path in a trellis $T$ we get the label sequence $S(T)$ defined by the trellis.

A trellis $T$ is said to be linear if there is a vertex labeling of its vertices such that the label sequence $S(T)$ is a vector space. It is easy to check that the BCJR trellis is linear. We denote the set of label sequences corresponding to the rows of $G$ with parity check matrix $H$ by $G_H$. 

Example – (4, 2) binary Code

\[ G = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \quad H = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \]
Bounds on state complexity

Let $C$ be an $(n, k)$ binary linear block code. Then the maximum state complexity which is $\log_2$ (maximum state cardinality), of a minimal trellis $T$ for $C$ is upper bounded by $s \leq \min(k, n - k)$. (Wolf 1978). This bound is attained by some codes. Thus trellis state space size can be exponential in the code parameters.

A lower bound (Ekroot et al, 1996) is:

\[ s \geq \lceil k(d - 1)/n \rceil = \lceil R(d - 1) \rceil \]

where $R$ is the rate of the code and $d$ its minimum distance.
Decoding on Trellises

- Maximum-likelihood decoding on a trellis is equivalent to finding the codeword closest to the received sequence measured in terms of a soft decision metric.

- The decoder uses the received vector $r$ to determine which codeword was transmitted. It forms an estimate $\hat{x}$ of the codeword $x$ that was transmitted. A decoding error occurs if $x \neq \hat{x}$. The maximum likelihood decoding rule is to decode the received sequence $r$ to codeword $x_m$ whenever $p(r/x_m) \geq p(r/x_l)$ for all $l \neq m$, where $p(r/x_m)$ is the conditional probability of $r$ given $x_m$.

- The Viterbi soft decision algorithm is used for decoding
• This is essentially a shortest path algorithm and has complexity linear in the size of the trellis.

• Is it possible to reduce this complexity?
Tail-biting trellises

A tail-biting trellis $T = (V, E, \mathbb{F}_2)$ of depth $n$ is an edge-labeled directed graph with the property that the set $V$ can be partitioned into $n$ vertex classes

$$V = V_0 \cup V_1 \cup \cdots \cup V_{n-1}$$

such that every edge in $T$ is labeled with a symbol from $\mathbb{F}_2$, and begins at a vertex of $V_i$ and ends at a vertex of $V_{i+1 \mod n}$, for some $i \in \{0, 1, \ldots, n - 1\}$. The time axis is circular which means that $V_0 = V_n$. 
Code represented by a tail-biting trellis

• The set of indices $\mathcal{I} = \{0, 1, \ldots, n - 1\}$ for the partition are the time indices. We identify $\mathcal{I}$ with $\mathbb{Z}_n$, the residue classes of integers modulo $n$.

• Every cycle of length $n$ in $T$ starting at a vertex of $V_0$ defines a vector $(a_1, a_2, \ldots, a_n) \in \mathbb{F}_2$ which is an edge-label sequence. We assume that every vertex and every edge in the tail-biting trellis lies on some cycle.

• The trellis $T$ is said to represent a block code $\mathcal{C}$ over $\mathbb{F}_2$ if the set of all edge-label sequences in cycles of $T$ beginning at some vertex in $V_0$ is equal to $\mathcal{C}$. Any vector that begins at time index 0 and ends at time index 0 but which is not a cycle is called a semicodeword.
Examples of Tail-biting Trellises

Figure 3: A tail-biting trellis for the code \{0000, 0110, 1001, 1111\}

Figure 4: A conventional trellis for the code \{0000, 0110, 1001, 1111\}
Figure 5: Conventional and tail-biting trellises for the binary \((7, 4)\) Hamming code.
Tail-biting trellises as Overlayed Sub-Automata

Figure 6: Minimal trellises for the subcode $C_0 = \{0000, 0110\}$ and coset $C_1 = \{1001, 1111\}$
The Product Construction of Tail-Biting Trellises

Definition. A circular span of vector $c$ is the smallest interval $[i, j]$, $i, j \in \{1, 2, \ldots, n\}$ that contains all the non-zero positions of $c$ with $i > j$. For example, $1010001$ has circular span $[7, 3]$. It also has circular span $[3, 1]$.

Thus while linear spans are unique to a vector, circular spans are not. They depend on the run of consecutive zeros chosen to define the span. There is an unique elementary trellis for each vector with a specified circular span.

Example

\[ G = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \quad [2, 3] \quad [4, 1] \]

A KV product trellis for the \((4, 2)\) linear code.
The Koetter-Vardy Theorem (2003)

Every linear biproper tail-biting trellis can be constructed as a trellis product of elementary trellises corresponding to the $k$ rows of a generator matrix $G$ with spans such that no two spans begin or end in the same position. The generator matrix is of the form $G = \begin{bmatrix} G_l \\ G_c \end{bmatrix}$ where $G_l$ has all the rows of linear span and $G_c$ has all the rows of circular span.

The generator matrix in the above form specifies a coset decomposition. The coset leaders are the vectors in the subspace generated by $G_c$ and the subgroup $G_0$ is the subspace generated by $G_l$. The number of vertices in $V_0$ is $2^{r_c}$ where $r_c$ is the number of rows in $G_c$. 
The BCJR Construction of Tail-Biting Trellises

A. Nori and P. Shankar, 2006

Let $C$ be an $(n, k)$ linear block code with generator matrix $G$ having rows $g_1, g_2, \ldots, g_k$, each of which is a $1 \times n$ vector and parity check matrix $H = [h_1 \ h_2 \cdots \ h_n]$, having columns $h_i$ each of which is a $(n-k) \times 1$ vector.

The tail-biting BCJR specification includes an $(n-k) \times 1$ displacement vector $d_{g_i}$ with every generator row $g_i \in G$.

The displacement vector $d_c$ for any codeword $c \in C$ is defined as follows.

$$d_c = \sum_{i=1}^{k} \alpha_i d_{g_i}, \text{ where } c = \sum_{i=1}^{k} \alpha_i g_i, \ \alpha_i \in \mathbb{F}_2, \ g_i \in G$$
The displacement matrix

Specifically, the set of displacement vectors for an $k \times n$ generator matrix $G$ is an arbitrary set of $k$ vectors each of which is a $((n - k) \times 1)$ vector. The set could be linearly independent or dependent, with repetitions allowed. The set of transposed displacement vectors forms a $k \times (n - k)$ matrix which we call the displacement matrix.

\[
G = \begin{bmatrix}
0 & 1 & 1 & 0 \\
1 & 0 & 0 & 1 \\
1 & 0 & 0 & 1 \\
\end{bmatrix}
\]

Let the displacement matrix $D = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$.

Therefore $d_{0110} = \binom{0}{0}$ and $d_{1001} = \binom{0}{1}$, and we have $d_{0000} = \binom{0}{0}$ and $d_{1111} = \binom{0}{1}$.
The BCJR Tail-Biting Trellis

Every codeword \( c = (c_1, c_2, \ldots, c_n) \in \mathcal{C} \) induces a sequence of states \( \{s_i\}_{i=0}^{n-1} \), each state being labeled by a \((n - k) \times 1\) vector as follows.

\[
    s_i = \begin{cases} 
    d_c & \text{if } i = 0 \\
    s_{i-1} + c_i h_i & \text{otherwise}
    \end{cases}
\]

There is an edge labeled \( a \in \mathbb{F}_2 \) from state \( s_{i-1} \) to state \( s_i \), \( 1 \leq i \leq n-2 \), if and only if

\[
    s_i = s_{i-1} + ah_i
\]
Example
The T-BCJR Construction and Coset Decomposition

All rows of $G$ with displacement vector equal to 0 are in the subgroup $G_0$ with respect to which cosets are computed or equivalently in the coset with coset leader $0$.

Any row $r$ of $G$ with a non-zero displacement is in a coset whose coset leader is the vector formed by some linear combination of $r$ with the rows of $G_0$ and which yields some 0 label. Such a vector is unique.
Dual Trellises

Theorem. [Forney 1988] The minimal conventional trellis $T = (V, E, \mathbb{F}_q)$ for a linear code $C$ of length $n$ and the minimal conventional trellis $T^\perp = (V^\perp, E^\perp, \mathbb{F}_q)$ for its dual code $C^\perp$ have identical state-complexity profiles.
Theorem. Let $C$ be an $(n, k)$ code with generator and parity check matrices $G$ and $H$ respectively. Let $G_{H,D}$ represent the set of label sequences obtained by executing the T-BCJR labeling algorithm on the rows of $G$ using $G, H$ and $D$. Let $T = \langle G_{H,D} \rangle$ represent $C$. Then the dual BCJR ($T-\text{BCJR}^\perp$) trellis $T^\perp = (V^\perp, E^\perp, F_2)$ representing the $(n, n-k)$ dual code $C^\perp$ is obtained as

$$T^\perp = \langle H_{G,D^t} \rangle$$

Theorem. Let $T$ be a minimal linear trellis, either conventional or tail-biting, for a linear code $C$. Then there exists a minimal linear dual trellis $T^\perp$ for the dual code $C^\perp$ such that the SCP of $T^\perp$ is identical to the SCP of $T$. 

(A.Nori and P. Shankar 2006)
Decoding on Tail-Biting Trellises

- Brute force algorithm is to Viterbi decode on each subtrellis separately. This offers no gains.

- Approximate algorithms which give near optimal results have complexity linear in the size of the tail-biting trellis. The maximum state space size of the tail-biting trellis can as low as the square root of the maximum state space complexity of the conventional trellis.
Decoding on Tail-Biting Trellises

The decoding algorithm is cast as a shortest path problem in which each path is associated with a *metric*, and the problem is to find a codeword path with minimum metric. The \( A^* \) algorithm is used to cut down the search space. It does so by using a node metric which is the sum of the length of the shortest path from the source to a node and an *underestimate* of the length of the shortest path from the node to the goal node to guide the search. The algorithm derives its advantage from the fact that if the estimates used are close to the actual values then the search space that yields the optimal path is greatly reduced.
Decoding

The algorithm proposed here is a variant of the $A^*$ algorithm, which at any given instant, is executing an $A^*$ algorithm on exactly one of the subtrellises, with perhaps suspended executions of the algorithm on a set of other subtrellises. The subtrellis on which the algorithm is currently executing, appears the best in its potential to deliver the minimal cost path. Estimates are derived from one pass over the tail-biting trellis. An approximate algorithm processes each node at most twice. On a binary symmetric channel it fails whenever the error is such that there exists a semicodeword $c_s$ such that for all $c$ in the code

$$wt(c_s + e) \leq wt(e) \leq (c + e)$$
Summary

- Trellises for linear block codes have interesting structural properties that are closely related to their algebraic properties.

- Soft-decision decoding is possible on trellises, and that is why they are used despite their high complexity.

- Tail-biting trellises also possess a rich algebraic structure.

- Tail-biting trellises can reduce the width of a trellis and hence decoder complexity substantially.
Thank You