To be presented at a conference in honor of P. G. Farrell's 60th birthday, Lancaster, England, January 26–28, 1998.

Draft of 29 Oct 1997 12:12 p.m.

How to Compute Weight Enumerators for Convolutional Codes

Robert J. McEliece*

Department of Electrical Engineering
California Institute of Technology
Pasadena, California 91103

rjm@systems.caltech.edu

Abstract.

Techniques for finding transfer functions for convolutional codes have long been known, but the methods that have appeared in print are, for the most part, more art than science. In this tutorial paper, we describe a precise and efficient algebraic method called the transfer matrix method. The transfer matrix method allows us not only to compute transfer functions, but also the weight enumerators for certain block codes which can be derived from the parent convolutional code, viz., the sequence of truncated and tailbiting codes. The transfer matrix method is easiest to apply if one uses a symbolic manipulation program like Mathematica, but even so, it is practical only for codes with relatively small degree.

* This work was supported by NSF grant no. NCR-9505975, AFOSR grant no. 5F49620-97-1-0313, and grant from Qualcomm, Inc.

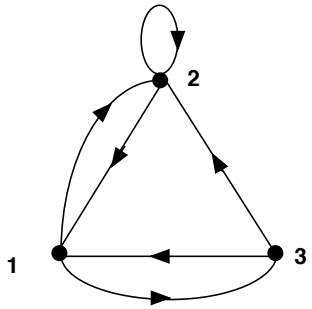

Figure 1. A simple digraph.

1. Digraphs and Trellises. The Transfer Matrix method.

In this section, we summarize the basic combinatorial and algebraic tools we will need to compute the weight enumerators described in Section 2. Much of the material in this section is taken from Stanley [1, Section 4.7].

A finite *directed graph* or *digraph* D consists of a finite vertex set V and a finite edge set E, where $E \subseteq V \times V$. If $e = (\alpha, \beta) \in E$, then e is said to be a *directed edge* from α to β, and to have *initial vertex* α, denoted $\text{init}(e) = \alpha$, and *final vertex* β, denoted $\text{fin}(e) = \beta$. Figure 1 shows a simple digraph with vertex set $V = \{1, 2, 3\}$ and edge set $E = \{(1,2), (1,3), (2,1), (2,2), (3,1), (3,2)\}$.

A *path of length K from u to v* in D is a sequence of K edges $e_1 e_2 \cdots e_K$, such that $\text{init}(e_1) = u$, $\text{fin}(e_K) = v$, and $\text{fin}(e_i) = \text{init}(e_{i+1})$, for $1 \leq i < K$. For example, in the digraph of Figure 1, $((1,2), (2,2), (2,1), (1,3))$ is a path of length 4 from 1 to 3. A path of length K can also be represented by a sequence of $K+1$ vertices $v_0 v_1 \cdots v_K$, where $(v_{i-1}, v_i) = e_i$. For example the path $((1,2), (2,2), (2,1), (1,3))$ in Figure 1 can also be represented by the vertex sequence $(1, 2, 2, 1, 3)$.

Now let $w : E \mapsto R$ be a *weight function* on the edge set E with values in some commutative semiring R. If $P = e_1 e_2 \cdots e_K$ is a path, then the *weight* of P is defined to be $W(P) = w(e_1) w(e_2) \cdots w(e_K)$. For example, in Figure 1, if we assume that all edges have weight 1, the path $((1,2), (2,2), (2,1), (1,3))$ also has weight 1.

A digraph D with a weight function on its edges can be represented a square matrix A, called the *incidence matrix* for D, whose rows and columns are indexed by V. If $e = (\alpha, \beta) \in E$, then the (α, β) entry of A is equal to $w(e)$.[1] If there is no edge from α to β, the (α, β) entry of A is equal to 0. For example, the digraph of Figure 1 can be

[1] If we allow multiple edges between pairs of vertices, the corresponding matrix entry is defined to be the sum of the weights of all such edges.

represented by the incidence matrix

$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix} \end{array}. \tag{1.1}$$

A *trellis* $T = (V, E)$ *of width* N is a finite directed graph of a special kind. The *vertex set* V is the disjoint union of $N + 1$ finite sets $V_0, \ldots, V_N$:

$$V = V_0 \cup V_1 \cup \cdots \cup V_N.$$

The *edge set* E is the disjoint union of N finite sets $E_1, \ldots, E_N$, where $E_i \subseteq V_{i-1} \times V_i$:

$$E = E_1 \cup E_2 \cup \cdots \cup E_N.$$

Thus an edge $e \in E_i$ is of the form $e = (\alpha, \beta)$ with $\alpha \in V_{i-1}$, and $\beta \in V_i$. For example, Figure 2 shows a simple trellis with

$$V_0 = \{a, b, c\},\ V_1 = \{d, e\},\ V_2 = \{f, g, h, k\}$$

and

$$E_1 = \{(a, d), (b, d), (b, e), (c, e)\},\ E_2 = \{(d, f), (d, g), (d, k), (e, f), (e, h), (e, k)\}.$$

Note that the trellis of Figure 2 also has edge weights.

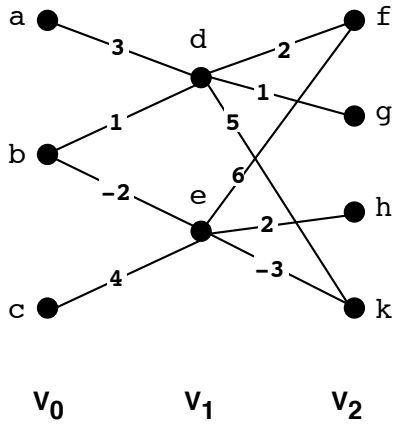

Figure 2. A simple weighted trellis of width 2.

A trellis T of width N with weighted edges can be represented by a sequence of N matrices $A_1, A_2, \ldots, A_N$. The ith matrix A_i has dimensions $|V_{i-1}| \times |V_i|$, with rows indexed by V_{i-1} and columns indexed by V_i. If $e = (\alpha, \beta) \in E_i$, with $\alpha \in V_{i-1}$ and $\beta \in V_i$, then the (α, β) entry of A_i is equal to $w(e)$. If there is no edge from α to β, the (α, β) entry of A_i is equal to 0. For example, the trellis of Figure 2 can be represented by the matrices A_1 and A_2, given by

$$A_1 = \begin{array}{c} \\ a \\ b \\ c \end{array} \begin{array}{c} \begin{array}{cc} d & e \end{array} \\ \begin{pmatrix} 3 & 0 \\ 1 & -2 \\ 0 & 4 \end{pmatrix} \end{array}, \quad A_2 = \begin{array}{c} \\ d \\ e \end{array} \begin{array}{c} \begin{array}{cccc} f & g & h & k \end{array} \\ \begin{pmatrix} 2 & 1 & 0 & 5 \\ 6 & 0 & 2 & -3 \end{pmatrix} \end{array}. \tag{1.2}$$

We shall call $A_1, A_2, \ldots, A_N$ the *matrix representation* of the corresponding weighted trellis. Conversely, if $A_1, A_2, \ldots, A_N$ is a sequence of matrices over a semiring R, and if the matrices are compatible, i.e., if for each $1 \leq i < N$, the number of columns of A_i is the same as the number of rows of A_{i+1}, we can construct a weighted trellis of width N from the sequence $A_1, A_2, \ldots, A_N$ in a straightforward way. The trellis constructed this way is called the trellis *afforded* by the sequence $A_1, A_2, \ldots, A_N$.

If the matrices $A_1, A_2, \ldots, A_N$ are compatible in the sense just described, then the matrix product $A = A_1 A_2 \cdots A_N$ is defined. This suggests that the matrix product A may have combinatorial significance. This is indeed the case, as we shall now see.

If $u \in V_0$ and $v \in V_N$, we define the *flow* from u to v, denoted $\phi_T(u, v)$, to be the sum of the weights of all paths in T from u to v. The combinatorial quantity $\phi_T(u, v)$ is related to the matrix representation of T by the following simple result.

1.1 Theorem. *The flow $\phi_T(u, v)$ is equal to the (u, v)th entry of the product matrix*

$$A = A_1 A_2 \cdots A_N.$$

Proof: (See [1,Theorem 4.7.1.]) By definition of matrix multiplication,

$$A[i, j] = \sum A_1[i, i_1] A_2[i_1, i_2] \cdots A_N[i_{N-1}, j],$$

where the sum is over all possible sequences $(i_1, i_2, \ldots, i_{N-1})$ of indices in the matrices $A_1, \ldots, A_N$. This sum is zero unless there is a path of the form $(i, i_1, i_2, \ldots, i_{N-1}, j)$ from i to j in the trellis afforded by the sequence $A_1, \ldots, A_N$. If such a path exists, then the sum is equal to the sum of the weights of all such paths, which is what the theorem asserts. ∎

For example, the product of the matrices A_1 and A_2 in (1.2) is

$$A = \begin{array}{c} \\ a \\ b \\ c \end{array} \begin{array}{c} \begin{array}{cccc} f & g & h & k \end{array} \\ \begin{pmatrix} 6 & 3 & 0 & 15 \\ -10 & 1 & -4 & 11 \\ 24 & 0 & 8 & -12 \end{pmatrix} \end{array}.$$

As a partial check of Theorem 1.1 in this case, we note that in the trellis of Figure 2, there are no paths from a to h, so that $\phi_T(a,h) = 0$ which is also the (a,h) entry of A. Similarly, there are two paths from b to f, viz., (b,d,f) and (b,e,f), with path weights $1 \cdot 2 = 2$ and $-2 \cdot 6 = -12$ respectively. Thus $\phi_T(b,f) = -10$, which is indeed the (b,f) entry of A.

A common way to construct a trellis of width N is what we might call the *repeated concatenation* of N copies of a fixed weighted digraph. Thus let $D = (V,E)$ be a weighted digraph, and let B be its incidence matrix. Then the width N trellis constructed from D is the trellis afforded by the sequence $A_1, \ldots, A_N$, where $A_1 = A_2 = \cdots = A_N = B$. To distinguish between vertices at different depths in the trellis constructed this way, we introduce a "time index" t. Thus in the width-N trellis constructed from $D = (V,E)$, the vertex set is $V \times \{0,1,\ldots,N\}$, and the edge set E_i is

$$E_i = \{((\alpha, i-1),(\beta,i)) : (\alpha,\beta) \in E\}.$$

For example, if we start with the digraph in Figure 1, whose incidence matrix is given by (1.1), we can construct the width-4 trellis shown in Figure 3, where the "time index" is shown along the bottom.

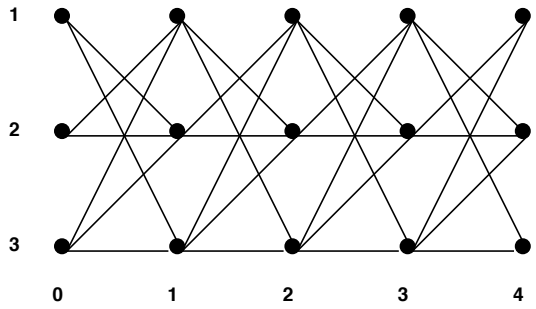

Figure 3. The width-4 trellis
constructed from the digraph of Figure 1.

We conclude this section with what Stanley [1] calls the *transfer-function* theorem for weighted digraphs. Let D be a weighted digraph with incidence matrix A, and for $N \geq 0$ let $A_{ij}(N)$ denote the sum of the weights of all paths in D of length N from v_i to v_j.[2] The quantities $A_{ij}(N)$ can be viewed as the entries of an $M \times M$ matrix $A(N)$. If we form the matrix generating function

$$F_G(z) = \sum_{N \geq 0} A(N) z^N, \tag{1.3}$$

we have the following basic result.

[2] We define $A_{ij}(0)$ to be 1 if $i = j$ and 0 if $i \neq j$.

1.2 Theorem. *(The transfer-function theorem) The generating function $F_G(z)$ is given by*

$$F_G(z) = (I - zA)^{-1}, \tag{1.4}$$

where in (1.4), I denotes the $M \times M$ identity matrix.

Proof: First note that the quantity $A_{ij}(N)$ is the sum of all paths from $(v_i, 0)$ to (v_j, N) in the trellis of width N constructed from D. Thus by Theorem 1.1, $A_{ij}(N)$ is the (i,j)th entry in the matrix A^N, i.e., $A(N) = A^N$. Thus

$$\begin{aligned} F_G(z) &= \sum_{N \geq 0} A(N) z^N \\ &= \sum_{N \geq 0} A^N z^N \\ &= (I - zA)^{-1}, \end{aligned}$$

using the "sum of a geometric series" theorem for matrices. ▪

2. Convolutional Codes, State Diagrams, and Incidence Matrices..

In this section we briefly review the state-space approach to convolutional codes, and state the counting problems we intend to solve.

An (n, k, m) *convolutional encoder* is a linear sequential device which maps a sequence of k-dimensional *information*, or *input blocks* $\boldsymbol{\alpha}_0, \boldsymbol{\alpha}_1, \ldots$ over the finite field $GF(2)$ into a sequence of n-dimensional *output*, or *code blocks* $\boldsymbol{\beta}_0, \boldsymbol{\beta}_1, \ldots$. The encoder also passes through a sequence of internal m-dimensional *state vectors*, or simply *states* $\boldsymbol{\sigma}_0, \boldsymbol{\sigma}_1 \ldots$. The jth code block $\boldsymbol{\beta}_j$ is a linear function of the jth input block $\boldsymbol{\alpha}_j$, and also of the jth state $\boldsymbol{\sigma}_j$. The formal description of the encoder's operation is this: the initial state $\boldsymbol{\sigma}_0$ is arbitrary and for $j \geq 0$,

$$\boldsymbol{\sigma}_{j+1} = \boldsymbol{\sigma}_j \mathcal{A} + \boldsymbol{\alpha}_j \mathcal{B} \tag{2.1}$$

$$\boldsymbol{\beta}_j = \boldsymbol{\sigma}_j \mathcal{C} + \boldsymbol{\alpha}_j \mathcal{D}, \tag{2.2}$$

where the four matrices $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$, and $\mathcal{D}$ have entries from $GF(2)$ and dimensions

$$\begin{aligned} \mathcal{A} &: m \times m \\ \mathcal{B} &: k \times m \\ \mathcal{C} &: m \times n \\ \mathcal{D} &: k \times n. \end{aligned} \tag{2.3}$$

The (n, k, m) *convolutional code* associated with such an encoder is then defined to be the set of all possible output sequences (or *codewords*) $(\boldsymbol{\beta}_0, \boldsymbol{\beta}_1, \ldots)$ which can be produced form the encoder, starting with the initial state $\boldsymbol{\sigma}_0 = \mathbf{0}$.

For example, consider the $(2,1,2)$ encoder shown in Figure 4. This encoder has input α, output $\boldsymbol{\beta} = (\beta_1, \beta_2)$ and state $\boldsymbol{\sigma} = (\sigma_1, \sigma_2)$. The next state will be $\boldsymbol{\sigma}' = (\alpha+\sigma_1+\sigma_2, \sigma_1)$, and the output $\boldsymbol{\beta}$ is given by $(\beta_1, \beta_2) = (\alpha, \alpha + \sigma_1)$, so the four matrices $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$, and $\mathcal{D}$ are, in this case,

$$\mathcal{A} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}, \quad \mathcal{B} = \begin{pmatrix} 1 & 0 \end{pmatrix}$$
$$\mathcal{C} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \quad \mathcal{D} = \begin{pmatrix} 1 & 1 \end{pmatrix}.$$

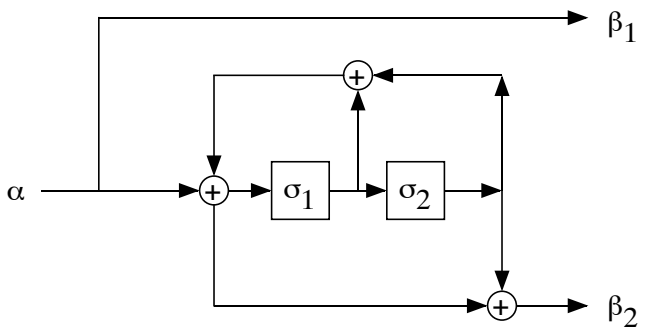

Figure 4. A simple $(2,1,2)$ convolutional encoder

Associated with each convolutional encoder is a corresponding *state diagram*, which is a weighted digraph of the type discussed in Section 1. The state diagram has 2^m vertices, one for each of the 2^m states of the encoder. For each state $\boldsymbol{\sigma}$ and input block $\boldsymbol{\alpha}$, there is a doubly-labelled edge from $\boldsymbol{\sigma}$ to $\boldsymbol{\sigma}'$:

$$\boldsymbol{\sigma} \xrightarrow[\boldsymbol{\beta}]{\boldsymbol{\alpha}} \boldsymbol{\sigma}' \tag{2.4}$$

where

$$\boldsymbol{\sigma}' = \boldsymbol{\sigma}\mathcal{A} + \boldsymbol{\alpha}\mathcal{B}$$
$$\boldsymbol{\beta} = \boldsymbol{\sigma}\mathcal{C} + \boldsymbol{\alpha}\mathcal{D}.$$

Thus the state diagram has 2^m vertices and 2^{m+k} edges. Figure 5 shows the state diagram for the encoder shown in Figure 4.

A semi-infinite *codepath* in the state diagram is a sequence of edges in the state diagram, say $e_1 e_2 \cdots$, where for each index $i = 1, 2, \ldots$, $\mathrm{fin}(e_i) = \mathrm{init}(e_{i+1})$. Such a codepath can be represented as follows:

$$\boldsymbol{\sigma}_0 \xrightarrow[\boldsymbol{\beta}_0]{\boldsymbol{\alpha}_0} \boldsymbol{\sigma}_1 \xrightarrow[\boldsymbol{\beta}_1]{\boldsymbol{\alpha}_1} \boldsymbol{\sigma}_2 \xrightarrow[\boldsymbol{\beta}_2]{\boldsymbol{\alpha}_2} \cdots \tag{2.5}$$

The codepath in (2.5) is said to have *initial state* $\boldsymbol{\sigma}_0$ and to be *generated by the input sequence* $(\boldsymbol{\alpha}_0, \boldsymbol{\alpha}_1, \ldots)$. The path is said to *produce the output sequence* $(\boldsymbol{\alpha}_0, \boldsymbol{\alpha}_1, \ldots)$. The

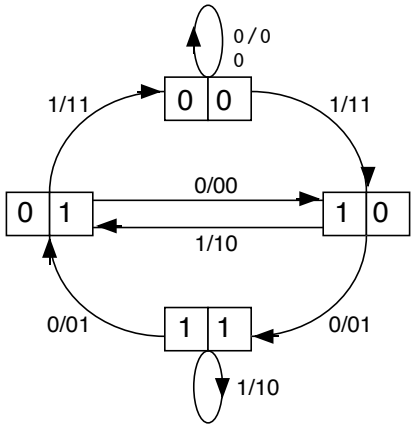

Figure 5. The state diagram for the encoder of Figure 4.

convolutional code associated with the encoding matrices $(\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D})$ is thus the set of all possible output sequences produced by semi-infinite codepaths of the form (2.5) which have initial state $\boldsymbol{\sigma}_0 = \mathbf{0}$.

A *finite codepath of length* N is a codepath of the form

$$\boldsymbol{\sigma}_0 \xrightarrow[\boldsymbol{\beta}_0]{\boldsymbol{\alpha}_0} \boldsymbol{\sigma}_1 \xrightarrow[\boldsymbol{\beta}_1]{\boldsymbol{\alpha}_1} \boldsymbol{\sigma}_2 \xrightarrow[\boldsymbol{\beta}_2]{\boldsymbol{\alpha}_2} \cdots \boldsymbol{\sigma}_{N-1} \xrightarrow[\boldsymbol{\beta}_{N-1}]{\boldsymbol{\alpha}_{N-1}} \boldsymbol{\sigma}_N. \tag{2.6}$$

In this paper we shall count finite codepaths of several types. Here are the four basic types we shall consider.

• The *closed codepaths*, i.e., the codepaths whose initial and final states are the same.

• The 0-*closed codepaths*, i.e., the codepaths whose initial and final states are both equal to $\mathbf{0}$.

• The *molecular codepaths*, i.e., the 0-closed codepaths which do not use the *zero edge* $\mathbf{0} \xrightarrow[\mathbf{0}]{\mathbf{0}} \mathbf{0}$.

• The *atomic codepaths*, i.e., the molecular codepaths which do not enter the zero state, except at the beginning and end.

We shall classify these codepaths according to two, and sometimes three, numerical attributes: the *length*, the *output weight*, and (sometimes) the *input weight*.[3] We shall do this classification using the method of generating functions, which are sometimes called

[3] It is unfortunate that the term *weight* has two different meanings in this paper. The weight of a path in a digraph is of course a very different thing from the weight of a vector or codeword. Both uses are well-established, however, and we have chosen to risk ambiguity rather than introduce new terminology.

weight enumerators in this context. For example, we if we denote by $w_{N,i,j}$ the number of 0-closed paths of length N, input weight i and output weight j, we will seek a closed-form expression for the three-variable generating function

$$W(x,y,z) = \sum_{N,i,j} w_{N,i,j} x^i y^j z^N.$$

The basic tools we use to count these codepaths are the transfer matrix theorem method described in Section 1 and xy-labelled and the y-labelled state diagrams of the encoder, and the corresponding incidence matrices.

The xy-weight state diagram of the code (also called the xy-state diagram) is the diagram obtained by replacing the label $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ on each edge of the state diagram (see (2.4)) with the label $x^{|\boldsymbol{\alpha}|} y^{|\boldsymbol{\beta}|}$, where $|\boldsymbol{x}|$ denotes the weight of the vector $\boldsymbol{x}$. Thus the exponent of x represents the input weight, and the exponent of y represents the output weight, of the corresponding state transition, so that the weight of a codepath in the xy-state diagram is of the form $x^{w_I} y^{w_O}$, where w_I is the input weight, and w_O is the output weight, of the path. Similarly, the y-weight state diagram (also called the y-state diagram) is obtained by replacing the $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ label with $y^{|\boldsymbol{\beta}|}$, so that the weight of a path in the y-state diagram is simply y^{w_O}. In Figures 6 and 7, we see the xy-weight and y-weight state diagrams for the encoder of Figure 4.

For calculation purposes, the xy- and y-weight state diagrams can be be represented by the corresponding incidence matrices. For example, the xy-incidence matrix corresponding to the xy-state diagram of Figure 6 is

$$A(x,y) = \begin{array}{c} \\ 00 \\ 10 \\ 01 \\ 11 \end{array} \begin{array}{c} \begin{array}{cccc} 00 & 10 & 01 & 11 \end{array} \\ \begin{pmatrix} 1 & xy^2 & 0 & 0 \\ 0 & 0 & xy & y \\ xy^2 & 1 & 0 & 0 \\ 0 & 0 & y & xy \end{pmatrix} \end{array}, \tag{2.7}$$

and the y-incidence matrix corresponding to the y-state diagram of Figure 7 is

$$A(y) = \begin{array}{c} \\ 00 \\ 10 \\ 01 \\ 11 \end{array} \begin{array}{c} \begin{array}{cccc} 00 & 10 & 01 & 11 \end{array} \\ \begin{pmatrix} 1 & y^2 & 0 & 0 \\ 0 & 0 & y & y \\ y^2 & 1 & 0 & 0 \\ 0 & 0 & y & y \end{pmatrix} \end{array}. \tag{2.8}$$

Finally, in our study of the molecular and atomic codepaths, we will need the *expurgated* xy incidence matrix for the code, denoted $A^{\bullet}(x,y)$. This matrix is identical to its counterpart $A(x,y)$, except that the $(0,0)$ entry is reduced by 1, which is the contribution to $A(x,y)$ of the *zero edge*

$$\mathbf{0} \xrightarrow[\mathbf{0}]{\mathbf{0}} \mathbf{0}.$$

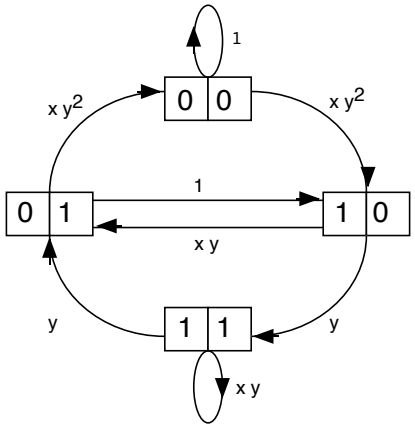

Figure 6. The xy-state diagram for the encoder of Figure 4.

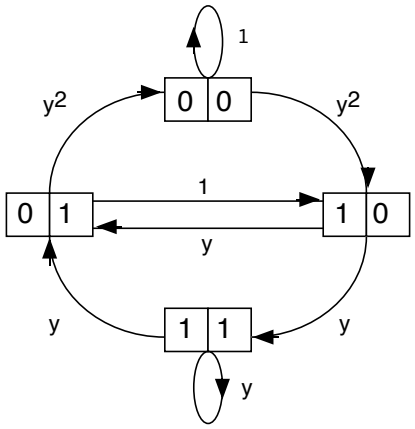

Figure 7. The y-state diagram for the encoder of Figure 4.

For example, for the encoder of Figure 4,

$$A^{\bullet}(x,y) = \begin{array}{c} \\ 00 \\ 10 \\ 01 \\ 11 \end{array} \begin{array}{c} \begin{array}{cccc} 00 & 10 & 01 & 11 \end{array} \\ \begin{pmatrix} 0 & xy^2 & 0 & 0 \\ 0 & 0 & xy & y \\ xy^2 & 1 & 0 & 0 \\ 0 & 0 & y & xy \end{pmatrix} \end{array}. \tag{2.9}$$

3. Molecular and Atomic Paths. The Transfer Function.

In this section we will find the xy-weight enumerators for the set of molecular and atomic codepaths. Specifically, we will investigate the generating functions

$$V^\bullet(x,y,z) = \sum_{N\geq 0} V_N^\bullet(x,y) z^N \tag{3.1}$$

$$W^\bullet(x,y,z) = \sum_{N\geq 0} W_N^\bullet(x,y) z^N, \tag{3.2}$$

where in (3.1), $V_N^\bullet(x,y)$ denotes the xy weight enumerator for the molecular codepaths of length N, and $W_N^\bullet(x,y)$ denotes the xy weight enumerator for the atomic codepaths of length N. The trivariate generating function $W^\bullet(x,y,z)$ is often called the *transfer function* for the code, and is useful for estimating the code's performance on a memoryless channel [2, Sec. 4.3].

3.1 Theorem. *We have*

$$V^\bullet(x,y,z) = \left[(I - zA^\bullet(x,y))^{-1}\right]_{0,0} \tag{3.3}$$

$$W^\bullet(x,y,z) = 1 - \frac{1}{\left[(I - zA^\bullet(x,y))^{-1}\right]_{0,0}}. \tag{3.4}$$

Proof: Since the molecular codepaths are, by definition, the 0-closed codepaths which do not use the zero edge, the trellis for the xy-weight enumerator for the molecular codepaths of length N is the one afforded by the matrix sequence

$$(A^\bullet(x,y), \ldots, A^\bullet(x,y)) \qquad (N \text{ terms}),$$

it follows from the Theorem 1.1 that the weight enumerator $V_N^\bullet(x,y)$ is given by the formula

$$V_N^\bullet(x,y) = \left[A^\bullet(x,y)^N\right]_{0,0}.$$

Thus we have

$$\begin{aligned} V^\bullet(x,y,z) &= \sum_{N\geq 0} V_N^\bullet(x,y) z^N \\ &= \sum_{N\geq 0} \left[A^\bullet(x,y)^N\right]_{0,0} z^N \\ &= \left[\sum_{N\geq 0} A^\bullet(x,y)^N z^N\right]_{0,0} \\ &= \left[(I - zA^\bullet(x,y))^{-1}\right]_{0,0}, \end{aligned}$$

which proves (3.3).

To prove (3.4), note that each molecular codepath can be "factored" uniquely into a concatenation of a finite number (possibly zero) of atomic codepaths. Thus, since $W^\bullet(x,y,z)$ denotes the enumerator for the atomic codepaths, we have

$$\begin{aligned} V^\bullet(x,y,z) &= 1 + W^\bullet(x,y,z) + W^\bullet(x,y,z)^2 + W^\bullet(x,y,z)^3 + \cdots \\ &= 1/(1 - W^\bullet(x,y,z)). \end{aligned}$$

Thus $W^\bullet(x,y,z) = 1 - 1/V^\bullet(x,y,z)$, which, given (3.3), proves (3.4). ▪

We conclude this section by illustrating the results for the encoder of Figure 4. Here $A^\bullet(x,y)$ is given by (2.9). Using *Mathematica*, or a similar symbolic manipulation program, it is then easy to apply (3.3) to find that

$$V^\bullet(x,y,z) = \frac{1 - xyz - xyz^2 + (-y^2 + x^2y^2)z^3}{1 - xyz - xyz^2 + (-y^2 + x^2y^2 - x^3y^5)z^3 + (-x^2y^6 + x^4y^6)z^4},$$

and

$$W^\bullet(x,y,z) = \frac{x^3y^5z^3 + (x^2y^6 - x^4y^6)z^4}{1 - xyz - xyz^2 + (-y^2 + x^2y^2)z^3}. \tag{3.5}$$

It follows then from the form of (3.5), that the individual weight enumerators $W_N^\bullet(x,y)$ in (3.2) satisfy the recursion

$$W_N^\bullet(x,y) = xyW_{N-1}^\bullet(x,y) + xyW_{N-2}^\bullet(x,y) + (y^2 - x^2y^2)W_{N-3}^\bullet(x,y) \qquad \text{for } N \geq 6. \tag{3.6}$$

We can obtain the first few terms of the expansion of $W^\bullet(x,y,z)$ in powers of z using long division:

$$W^\bullet(x,y,z) = x^3y^5z^3 + x^2y^6z^4 + (x^4y^6 + x^3y^7)z^5 + (\text{terms of order } z^6 \text{ and higher}),$$

which gives the entries in rows 3, 4, and 5 in Table 1. Given these three rows as initial conditions, using the recursion (3.6), we can obtain (by computer) as many rows in Table 1 as desired. (In Table 1, we have represented the function $W_N^\bullet(x,y)$ as $\sum_j w_{N,j}^\bullet(x)y^j$, where $w_{N,j}^\bullet$ is a polynomial in x.)

$N\backslash j$	**5**	**6**	**7**	**8**	**9**	**10**	**11**
0							
1							
2							
3	x^3						
4		x^2					
5		x^4	x^3				
6			$2x^3$	x^4			
7			x^5	x^2+2x^4	x^5		
8				$3x^4$	$2x^3+2x^5$	x^6	
9				x^6	$3x^3+3x^5$	$3x^4+2x^6$	x^7
$\vdots$							

Table 1. The polynomials $w^{\bullet}_{N,j}(x)$ for the encoder of Figure 4.

4. Truncated and Tailbiting Codes, Output Weight Only.

In this section we will find generating functions for the set of 0-closed and closed codepaths, which will give us the weight enumerators for the truncated and tailbiting[4] block codes derived from the parent convolutional code. Specifically, we will investigate the generating functions

$$W(y,z) = \sum_{N\geq 0} W_N(y) z^N \tag{4.1}$$

$$W^*(y,z) = \sum_{N\geq 0} W^*_N(y) z^N, \tag{4.2}$$

where in (4.1), $W_N(y)$ denotes the y weight enumerator for the 0-closed codepaths of length N, and $W^*_N(y)$ denotes the y weight enumerator for the set of closed codepaths of length N.

4.1 Theorem. *We have*

$$W(y,z) = \left[(I - zA(y))^{-1}\right]_{0,0} \tag{4.3}$$

$$W^*(y,z) = \mathrm{Tr}\left[(I - zA(y))^{-1}\right]. \tag{4.4}$$

[4] The Nth *truncated code* derived from a parent convolutional code is the set of codewords generated by all finite codepaths of length N whose initial and final states are both 0. The Nth *tailbiting* code is the set of codewords generated by all finite codepaths of length N whose initial and final states are the same.

Proof: Since the trellis for the y-weight enumerator for the closed codepaths of length N is the one afforded by the matrix sequence

$$(A(y), \ldots, A(y)) \qquad (N \text{ terms}),$$

it follows from Theorem 1.1 that the weight enumerator $W_N(y)$ is given by

$$W_N(y) = \left[A(y)^N\right]_{0,0}.$$

Thus we have

$$\begin{aligned} W(y,z) &= \sum_{N\geq 0} W_N(y) z^N \\ &= \sum_{N\geq 0} \left[A(y)^N\right]_{0,0} z^N \\ &= \left[\sum_{N\geq 0} A(y)^N z^N\right]_{0,0} \\ &= \left[(I - zA(y))^{-1}\right]_{0,0}, \end{aligned}$$

which proves (4.3). The result (4.4) is proved similarly.[5] ∎

The following result gives an elegant alternative method for calculating the enumerators $W_N^*(y)$.

4.2 Corollary. *We have*

$$\sum_{N\geq 1} W_N^*(y) z^N = -\frac{z\frac{\partial Q(y,z)}{\partial z}}{Q(y,z)}, \tag{4.5}$$

where $Q(y,z) = \det(I - zA(y))$. *(Note that the sum in (4.5) begins at* $N = 1$, *whereas the sum in (4.2) begins at* $N = 0$.*)*

Proof: This follows immediately from Corollary 4.7.3 in [1], which proves that in an arbitrary labelled digraph with incidence matrix A,

$$\sum_{N\geq 1} C_G(N) z^N = -\frac{zQ'(z)}{Q(z)},$$

where $C_G(N)$ denotes the sum of the weights of all closed paths of length N and $Q(z) = \det(I - zA)$. ∎

[5] In (4.4), the notation "Tr" represents the trace of a square matrix, i.e., the sum of the diagonal entries.

We conclude this section by illustrating the results for the encoder of Figure 4. Here $A(y)$ is given by (2.8). Using *Mathematica* and Theorem 4.1, we find that

$$W(y,z) = \frac{1 - yz - yz^2}{1 - (1+y)z - (-y+y^5)z^3}. \tag{4.6}$$

From (4.1) and (4.6), we see that the individual weight enumerators $W_N(y)$ satisfy the recursion

$$W_N(y) = (1+y)W_{N-1}(y) + (-y+y^5)W_{N-3}(y). \qquad \text{for } N \geq 3. \tag{4.7}$$

Expanding (4.6) in ascending powers of z, either by hand or by using *Mathematica*, we find

$$W(y,z) = 1 + z + z^2 + (\text{terms of order } z^3 \text{ and higher})$$

which gives the initial conditions

$$W_0(y) = W_1(y) = W_2(y) = 1,$$

which appear as the first three rows of Table 2. The remainder of Table 2 is then easily found using the recursion (4.7). (In Table 2, we have represented the function $W_N(y)$ as $\sum_j w_{N,j} y^j$.)

$N\backslash j$	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	$\cdots$
0	1																
1	1																
2	1																
3	1					1											
4	1					2	1										
5	1					3	3	1									
6	1					4	5	4	1		1						
7	1					5	7	8	5	1	3	2					
8	1					6	9	12	12	6	7	8	3				
9	1					7	11	16	20	17	17	19	15	4		1	
$\vdots$																	

Table 2. The numbers $w_{N,j}$ for the encoder of Figure 4.

Similarly, an application of Theorem 4.1 shows that

$$W^*(y,z) = \frac{4 - (3+3y)z - (-y+y^5)z^3}{1 - (1+y)z - (-y+y^5)z^3}. \tag{4.8}$$

From (4.2) and (4.8), we see that the individual weight enumerators $W_N^*(y)$ satisfy the same recursion as the $W_N(y)$'s, viz.,

$$W_N^*(y) = (1+y)W_{N-1}^*(y) + (-y+y^5)W_{N-3}^*(y), \qquad \text{for } N \geq 3. \tag{4.9}$$

Expanding (4.8) in ascending powers of z, either by hand or by using *Mathematica*, we find

$$W^*(y,z) = 4 + (1+y)z + (1+2y+y^2)z^2 + (\text{terms of order } z^3 \text{ and higher}),$$

i.e.,

$$\begin{aligned} W_0^*(y) &= 4 \\ W_1^*(y) &= 1+y \\ W_2^*(y) &= 1+2y+y^2 \end{aligned}$$

which gives the first three rows of Table 3. It is then easy to complete Table 3 using the recursion (4.9). (In Table 3, we have represented the function $W_N^*(y)$ as $\sum_j w_{N,j}^* y^j$.)

$N\backslash j$	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	$\cdots$
0	4																
1	1	1															
2	1	2	1														
3	1		3	1		3											
4	1		2	4	1	4	4										
5	1			5	5	6	10	5									
6	1			2	9	12	13	18	6		3						
7	1				7	21	21	29	28	7	7	7					
8	1				2	24	36	40	57	40	20	24	12				
9	1					18	48	63	81	100	72	54	54	18		3	
$\vdots$																	

Table 3. The numbers $w_{N,j}^*$ for the encoder of Figure 4.

If we use the alternative method of computing the $W_N^*(y)$'s given in Corollary 4.2, we find that the polynomial $Q(y,z) = \det(I - zA(y))$, where $A(y)$ is given by (2.8), is

$$Q(y,z) = 1 - (1+y)z - (-y+y^5)z^3. \tag{4.10}$$

Thus from Corollary 4.2, we have

$$\sum_{N\geq 1} W_N^*(y) z^N = -\frac{z\frac{\partial Q(y,z)}{\partial z}}{Q(y,z)} = \frac{(1+y)z + (-3y+3y^5)z^3}{1-(1+y)z-(-y+y^5)z^3}. \tag{4.11}$$

From (4.11), we again see that the recursion (4.9) holds, and so are led to the expansion

$$\sum_{N\geq 1} W_N^*(y)z^N = (1+y)z + (1+2y+y^2)x^2 + (1+3y^2+y^3+3y^5)z^3 + \cdots,$$

which gives (except for $N=0$) the same values for the $W_N^*(y)$'s as the method of Theorem 4.1.

5. Truncated and Tailbiting Codes, Input-Output Weight Enumerators.

In this section we restrict ourselves to convolutional codes with $k=1$. We will find xyz-generating functions for the set of 0-closed and closed codepaths, which will give us the input-output weight enumerators for the truncated and tailbiting block codes derived from the parent convolutional code (with $k=1$). Specifically, we will investigate the generating functions

$$W(x,y,z) = \sum_{N\geq m} W_N(x,y)z^N \tag{5.1}$$

$$W^*(x,y,z) = \sum_{N\geq m} W_N^*(x,y)z^N, \tag{5.2}$$

where in (5.1), $W_N(x,y)$ denotes the xy weight enumerator for the set of 0-closed codepaths of length N, and $W_N^*(x,y)$ denotes the xy weight enumerator for the set of closed codepaths of length N. We assume that the encoder's input is "turned off" after $N-m$ clock cycles and that the encoder is then forced into the appropriate terminal state during the final m clock cycles.

5.1 Theorem. *We have*

$$W(x,y,z) = z^m \left\{ (I - zA(x,y))^{-1} A(y)^m \right\}_{(0,0)} \tag{5.3}$$

$$W^*(x,y,z) = z^m \operatorname{Tr}\left\{ \Delta(x)(I - zA(x,y))^{-1} A^m(y) \right\}, \tag{5.4}$$

where in (5.4), $\Delta(x)$ is a diagonal matrix whose (s,s)th entry is $x^{|s|}$.

Proof: According to our assumption, for a truncated or a tailbiting code of length N, the encoder accepts inputs for the first $N-m$ times clock cycles, at which point the input is turned off for the remaining m clock cycles so that the trellis path can be properly terminated. Thus for computing $W_N(x,y)$ and $W_N^*(x,y)$, the appropriate trellis is the one afforded by the matrix sequence

$$A(x,y), A(x,y), \ldots, A(x,y), A(y), \ldots, A(y),$$

where there are $N-m$ terms equal to $A(x,y)$, and m terms equal to $A(y)$. It follows then from Theorem 1.1 that

$$W_N(x,y) = \left[A(x,y)^{N-m} A(y)^m \right]_{0,0}.$$

Thus we have

$$\begin{aligned} W(x,y,z) &= \sum_{N\geq m} W_N(x,y) z^N \\ &= \sum_{N\geq m} \left[A(x,y)^{N-m} A(y)^m\right]_{0,0} z^N \\ &= z^m \left[\sum_{N\geq m} z^{N-m} A(x,y)^{N-m} A(y)^m\right]_{0,0} \\ &= z^m \left[\left(\sum_{h\geq 0} A(x,y)^h z^h\right) A(y)^m\right]_{0,0} \\ &= z^m \left\{(I - zA(x,y))^{-1} A(y)^m\right\}_{(0,0)}, \end{aligned}$$

which proves (5.3).

To prove (5.3), we note that the first m information bits in the encoder for a tailbiting code of length N are used to select the initial state, at which point the encoder behaves like the encoder for the truncated code. Thus

$$W_N^*(x,y) = \sum_s x^{|s|} \left[A(x,y)^{N-m} A(y)^m\right]_{0,0}.$$

Thus we have

$$\begin{aligned} W^*(x,y,z) &= \sum_{N\geq m} W_N^*(x,y) z^N \\ &= \sum_{N\geq m} z^N \sum_s x^{|s|} \left[A(x,y)^{N-m} A(y)^m\right]_{0,0} \\ &= \sum_{N\geq m} z^N \operatorname{Tr}\left\{\Delta(x) A(x,y)^{N-m} A(y)^m\right\} \\ &= \operatorname{Tr}\left\{z^m \Delta(x) \left(\sum_{h\geq 0} A(x,y)^h z^h\right) A(y)^m\right\} \\ &= z^m \operatorname{Tr}\left\{\Delta(x)[I - zA(x,y)]^{-1} A(y)^m\right\}, \end{aligned}$$

which proves (5.3).∎

We conclude this section by illustrating the results for the encoder of Figure 4. Here $A(x,y)$ is given by (2.7), and $A(y)$ is given by (2.8). Using *Mathematica*, it is then easy to apply (5.3) to find that

$$W(x,y,z) = \frac{P(x,y,z)}{Q(x,y,z)},$$

where

$$P(x,y,z) = z^2 + (-xy + xy^5)z^3 + (-xy + x^2y^5 + xy^6 - x^2y^6)z^4 + (-y^2 + x^2y^2 + xy^6 - x^3y^6)z^5 \tag{5.5}$$

and

$$Q(x,y,z) = 1 + (-1 - xy)z + (xy - y^2 + x^2y^2 - x^3y^5)z^3 + (y^2 - x^2y^2 - x^2y^6 + x^4y^6)z^4. \tag{5.6}$$

Using *Mathematica*, we find that the first few terms in the expansion of $W(x,y,z)$ in powers of z are

$$W(x,y,z) = z^2 + (1 + xy^5)z^3 + (1 + (x + x^2)y^5 + xy^6)z^4 + (1 + (x + x^2 + x^3)y^5 + (2x + x^2)y^6 + x^2y^7)z^5 + \text{terms of order } z^6 \text{ and higher} \tag{5.7}$$

The remaining terms in the expansion of $W(x,y,z)$ can be obtained by noting that the form of the denominator in (5.6) implies that the individual xy-weight enumerators $W_N = W_N(x,y)$ satisfy the fourth-order recursion

$$W_N = (1 + xy)W_{N-1} + (-xy + y^2 - x^2y^2 + x^3y^5)W_{N-3} + (-y^2 + x^2y^2 + x^2y^6 - x^4y^6)W_{N-4} \qquad \text{for } N \geq 6. \tag{5.8}$$

Hence using the initial conditions provided by (5.7), viz.,

$$\begin{aligned} W_2(x,y) &= 1 \\ W_3(x,y) &= 1 + xy^5 \\ W_4(x,y) &= 1 + (x + x^2)y^5 + xy^6 \\ W_5(x,y) &= 1 + (x + x^2 + x^3)y^5 + (2x + x^2)y^6 + x^2y^7 \end{aligned}$$

and the recursion (5.8), we obtain the values shown in Table 4, which may easily be extended as far as desired. (In Table 4, we have represented the function $W_N(x,y)$ as $\sum_j w_{N,j}(x)y^j$, where $w_{N,j}$ is a polynomial in x.)

$N\backslash j$	**0**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
2	1							
3	1	x						
4	1	$x + x^2$	x					
5	1	$x + x^2 + x^3$	$2x + x^2$	x^2				
6	1	$x + x^2 + 2x^3$	$2x + 2x^2 + x^3$	$x + 3x^2$	x^3		x^4	
7	1	$x + x^2 + 3x^3$	$2x + 3x^2 + x^3 + x^4$	$x + 5x^2 + 2x^3$	$x + x^2 + 3x^3$	x^4	$2x^4 + x^5$	$x^3 + x^4$
$\vdots$								

Table 4. The polynomials $w_{N,j}(x)$ for the encoder of Figure 4.

Similarly, using (5.4), we find

$$W^*(x,y,z) = \frac{P^*(x,y,z)}{Q^*(x,y,z)}, \tag{5.9}$$

where

$$\begin{aligned} P^*(x,y,z) = {} & (1+2xy+x^2y^2)z^2 \\ & + (-3xy+2xy^2-2x^2y^2+xy^5+2x^2y^5)\ z^3 \\ & + (-xy-2xy^2+x^2y^2+xy^3-x^3y^3+x^3y^5+2xy^6-x^2y^6)\ z^4 \\ & + (-y^2+x^2y^2-xy^3+x^3y^3+x^2y^6-x^4y^6+x^3y^7-x^5y^7)\ z^5 \end{aligned} \tag{5.10}$$

and

$$\begin{aligned} Q^*(x,y,z) = {} & 1+(-1-xy)\ z+(xy-y^2+x^2y^2-x^3y^5)\ z^3 \\ & + (y^2-x^2y^2-x^2y^6+x^4y^6)\ z^4. \end{aligned} \tag{5.11}$$

Using *Mathematica*, we find that the first few terms in the expansion of $W^*(x,y,z)$ in powers of z are

$$\begin{aligned} W^*(x,y,z) = {} & (1+2xy+x^2y^2)z^2+(1+2xy^2+x^2y^2+x^3y^3+xy^5+2x^2y^5)z^3 \\ & + (1+2x^2y^2+xy^3+2x^2y^3+x^3y^3+x^4y^4 \\ & \qquad + xy^5+2x^2y^5+x^3y^5+2xy^6+2x^3y^6)z^4 \\ & + (1+2xy^3+2x^2y^3+x^3y^3+2x^2y^4+2x^3y^4 \\ & \qquad + x^4y^4+xy^5+2x^2y^5+2x^3y^5+x^5y^5+2xy^6+2x^2y^6 \\ & \qquad + 4x^3y^6+2x^4y^6+2x^2y^7+x^3y^7+2x^4y^7)z^5 \\ & + \text{terms of order } z^6 \text{ and higher} \end{aligned} \tag{5.12}$$

The remaining terms in the expansion of $W^*(x,y,z)$ can be obtained by noting that since the denominator $Q^*(x,y,z)$ is the same as $Q(x,y,z)$ (cf. (5.6) and (5.11)), the individual xy-weight enumerators $W_N^* = W_N^*(x,y)$ satisfy the same fourth-order recursion as $W_N(x,y)$:

$$\begin{aligned} W_N^* = {} & (1+xy)W_{N-1}^* + (-xy+y^2-x^2y^2+x^3y^5)W_{N-3}^* \\ & + (-y^2+x^2y^2+x^2y^6-x^4y^6)W_{N-4}^* \qquad \text{for } N \ge 6. \end{aligned} \tag{5.13}$$

Hence using the initial conditions provided by (5.7), viz.,

$$\begin{aligned} W_2^*(x,y) &= 1+2xy+x^2y^2 \\ W_3^*(x,y) &= 1+2xy^2+x^2y^2+x^3y^3+xy^5+2x^2y^5 \\ W_4^*(x,y) &= 1+2x^2y^2+xy^3+2x^2y^3+x^3y^3+x^4y^4+xy^5+2x^2y^5+x^3y^5+2xy^6+2x^3y^6 \\ W_5^*(x,y) &= 1+2xy^3+2x^2y^3+x^3y^3+2x^2y^4+2x^3y^4+x^4y^4+xy^5+2x^2y^5+2x^3y^5 \\ & \quad + x^5y^5+2xy^6+2x^2y^6+4x^3y^6+2x^4y^6+2x^2y^7+x^3y^7+2x^4y^7, \end{aligned}$$

and the recursion (5.13), we can extend this series as far as desired. (We omit any further terms, however, since e.g. $W_6^*(x,y)$ has 26 terms.)

References.

1. R. P. Stanley, *Enumerative Combinatorics, vol. I.* Monterey, Calif.: Wadsworth & Brooks/Cole, 1986.
2. A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding.* New York: McGraw-Hill, 1979.