Non-commutative convolutional codes over the infinite dihedral group

Abstract

Classic convolutional codes are defined as the convolution of a message and a transfer function over $\mathbb{Z}$. In this paper, we study convolutional codes over the infinite dihedral group D_∞. The goal of this study is to design convolutional codes with good and interesting properties and intended to be more resistant to code recognition. Convolution of two functions on D_∞ corresponds to the product of two polynomials in the noncommutative polynomial algebra $\mathbb{F}_2\{X,Y\}/(X^2-1, Y^2-1)$. We study this algebra and derive a criterion for an element to be regular. Such an element defines a transfer function. Moreover, we show how encoding over D_∞ can be represented by two classical convolutions over $\mathbb{Z}$, alternating according to the parity of the index of the input bits. Furthermore, we adapt the Viterbi algorithm to decode these codes using two different trellis. Finally, we show that these codes have performances similar to classic convolutional codes. Unfortunately, they are not more resistant to code recognition. However, under certain conditions, we get more optimal codes in terms of free distance than conventional convolutional codes, which can be a great asset for non-generic dedicated proprietary transmitter/receiver.

Keywords : Convolutional codes, coding theory, non-commutative polynomial algebra, non-commutative group, infinite dihedral group, transfer function, adapted Viterbi algorithm, free distance.

1 Introduction

Convolutional codes were introduced by Elias in 1955 [4]. A convolutional code of rate $\frac{1}{n}$ can be represented by the figure 1. In this figure, the bits u_i represent the bits of a message u. These bits enter into the shift register with a memory length m and are xored or not according to the bits $g_{l,j},\ 0 \leqslant l \leqslant m,\ 0 \leqslant j \leqslant n-1$. We obtain n bits $c_{i,j}$ at each time i, $0 \leqslant i < k$, that are serialized to produce the code word.

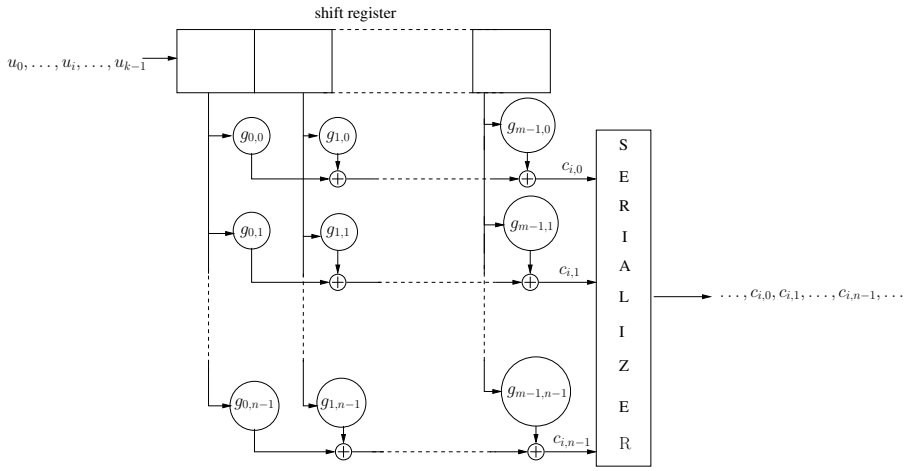

Figure 1: Classic convolutional code

The name of these codes comes from the convolution product of the input stream $u = (u_0, \ldots, u_{k-1})$ with the encoder's impulse responses $g_j = (g_{0,j}, \ldots, g_{m-1,j})$, $0 \leqslant j \leqslant n-1$:

$$c_{i,j} = \sum_{l=0}^{m-1} u_{i-l} g_{l,j}$$

We can transform this two-fold operation, convolution and serialization, by a single convolution with a transfer function τ. This can be seen most conveniently by using polynomial multiplication in the algebra $\mathbb{F}_2[X]$ of polynomials in X with coefficients in $\mathbb{F}_2$. Define polynomials $g_i(X)$ by :

$$\forall i \in \{0, \ldots, n-1\},\ g_i(X) = g_{0,i} + g_{1,i}X + \ldots + g_{m-1,i}X^{m-1},$$

and define the polynomial $\tau(X)$ by :

$$\tau(X) = g_0(X^n) + Xg_1(X^n) + \ldots + X^{n-1}g_{n-1}(X^n).$$

The single convolution of u with τ corresponds to the product $u(X^n)\tau(X)$ in $\mathbb{F}_2[X]$, where $u(X^n) = \sum_{i=0}^{k-1} u_i X^{ni}$. The convolution here is a convolution of functions defined on the set of natural numbers $\mathbb{N}$. In order to have a convolution over the group of integers $\mathbb{Z}$, we extend the functions by 0. The resulting convolution of functions on $\mathbb{Z}$ corresponds to the product $u(X^n)\tau(X)$ in the algebra $\mathbb{F}_2[X, X^{-1}]$.

In this paper, we replace the group $\mathbb{Z}$ by the infinite dihedral group D_∞. The goal of this work was initially to add cryptographic properties while maintaining good error correction properties, in order to have convolutional codes more resistant to code recognition [3] [6].

2 The infinite dihedral group and the associated non-commutative polynomial algebra

Recall that the finite dihedral group D_n of order $2n$ is defined by the presentation

$$D_n = \langle r, s \mid r^n = 1, s^2 = 1, srs = r^{-1} \rangle,$$

for any nonzero natural number n. The infinite dihedral group D_∞ is then naturally defined by the presentation

$$D_\infty = \langle r, s \mid s^2 = 1, srs = r^{-1} \rangle.$$

Equivalently, D_∞ is the semi-direct product $\mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z}$. Indeed, an element of D_∞ can be written as $s^\varepsilon r^a$, with $\varepsilon \in \{0, 1\} = \mathbb{Z}/2\mathbb{Z}$ and $a \in \mathbb{Z}$, in one and only one way. It follows that D_∞ can be identified with the set $\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$. The induced group law on $\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$ is given by :

$$\begin{aligned}
(a, 0) \circ (b, 0) &= (a + b, 0) \\
(a, 0) \circ (b, 1) &= (a + b, 1) \\
(a, 1) \circ (b, 0) &= (a - b, 1) \\
(a, 1) \circ (b, 1) &= (a - b, 0),
\end{aligned}$$

where $a, b \in \mathbb{Z}$. This is exactly the group law of the the semi-direct product $\mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z}$ on the set $\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$. We see, moreover, that the group $\mathbb{Z} = \mathbb{Z} \times \{0\}$ is a normal subgroup of index 2 in $\mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z}$.

Returning to the above presentation of D_∞, let us write $y = rs$ and $x = s$. Then we have the equivalent presentation

$$D_\infty = \langle x, y \mid x^2 = y^2 = 1 \rangle.$$

It shows that D_∞ can also be identified with the free product $\mathbb{Z}/2\mathbb{Z} * \mathbb{Z}/2\mathbb{Z}$ of the group $\mathbb{Z}/2\mathbb{Z}$ with itself.

Let $\mathbb{F}_2[D_\infty]$ be the $\mathbb{F}_2$-group algebra on D_∞, i.e., $\mathbb{F}_2[D_\infty]$ is the $\mathbb{F}_2$-vector space having the set D_∞ as a basis. Multiplication in $\mathbb{F}_2[D_\infty]$ is the $\mathbb{F}_2$-bilinear extension of the group law in D_∞. Note that $\mathbb{F}_2[D_\infty]$ is nothing

but the set of functions on D_∞ with values in $\mathbb{F}_2$ having finite support. The product in $\mathbb{F}_2[D_\infty]$ coincides with convolution of functions.

Since we want to think of $\mathbb{F}_2[D_\infty]$ as a noncommutative polynomial algebra, we write X instead of x, and Y instead of y. Then

$$\mathbb{F}_2[D_\infty] = \mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1),$$

where $\mathbb{F}_2\{X, Y\}$ denotes the $\mathbb{F}_2$-algebra of noncommutative polynomials in X and Y, and $(X^2 - 1, Y^2 - 1)$ is the two-sided ideal in $\mathbb{F}_2\{X, Y\}$ generated by $X^2 - 1$ and $Y^2 - 1$.

Since $r = yx$ and $s = x$ in D_∞, any element $P \in \mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$ can be written as

$$P = \sum_{(i,j)\in\mathbb{Z}\rtimes\mathbb{Z}/2\mathbb{Z}} c_{i,j} X^j (YX)^i,$$

with $c_{i,j} \in \mathbb{F}_2$, in one and only one way.

Observe that the sub-algebra $\mathbb{F}_2[XY, YX]$ of $\mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$ generated by XY and YX is commutative and isomorphic to the algebra $\mathbb{F}_2[T, T^{-1}]$, which incidentally is nothing else but the $\mathbb{F}_2$-group algebra $\mathbb{F}_2[\mathbb{Z}]$ of the group $\mathbb{Z}$. We can therefore consider $\mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$ as a right $\mathbb{F}_2[XY, YX]$-module, and as such it is free with basis $1, X$, as follows from what we have seen above. Explicitly this means the following. Let $P \in \mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$. Then there are $A, B \in \mathbb{F}_2[XY, YX]$ such that

$$P = A + XB$$

in $\mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$. Moreover, A and B are uniquely determined by P.

In order to have an error correcting code over the non-commutative group D_∞ defined by convolution, each code word must be the result of one and only one message by convolution, otherwise we can not decode. So convoluting by the transfer function must be injective. To put it otherwise, the transfer function must not be a right zero divisor in the ring $\mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$, i.e., it must be a regular element of $\mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$:

Definition 1. *Let R be a ring. An element P of R is a right regular element of R if :*

$$\forall C \in R,\ CP = 0 \Rightarrow C = 0$$

Theorem 1. *Any nonzero element of the subalgebra $\mathbb{F}_2[XY, YX]$ of $\mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$ is a right regular element of $\mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$.*

Proof. Let be $P \in \mathbb{F}_2[XY, YX]$ be nonzero. We show that P is a right regular element of $\mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$. Let $C \in \mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$ be such that $CP = 0$. As we have seen above, there are $A, B \in \mathbb{F}_2[XY, YX]$ such that $C = A + XB$. We have

$$0 + X0 = CP = (A + XB)P = AP + X(BP).$$

Since AP and BP are elements of $\mathbb{F}_2[XY, YX]$, one deduces that $AP = 0$ and $BP = 0$. Now, the algebra $\mathbb{F}_2[XY, YX]$ is isomorphic to the algebra $\mathbb{F}_2[T, T^{-1}]$ and is, therefore, integral. Since $P \neq 0$, one necessarily has $A = 0$ and $B = 0$, i.e. $C = 0$. □

Let us define an anti-automorphism

$$\begin{gathered}\mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1) \to \mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1) \\ P \mapsto \overline{P}\end{gathered}$$

by $\overline{X} = X$ and $\overline{Y} = Y$. This means that $\overline{PQ} = \overline{QP}$, for all $P, Q \in \mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$. In particular, we have $\overline{(XY)^n} = (YX)^n$ and $\overline{X(YX)^n} = X(YX)^n$, for all integers n. It follows that

$$\overline{A + XB} = \overline{A} + XB,$$

for all $A, B \in \mathbb{F}_2[XY, YX]$.

Since $P \mapsto \overline{\overline{P}}$ is an automorphism of $\mathbb{F}_2\{X, Y\}/(X^2 - 1, Y^2 - 1)$ fixing X and Y, it is the identity. Hence, one has $\overline{\overline{P}} = P$ for all $P \in \mathbb{F}_2\{X, Y\}/-(X^2 - 1, Y^2 - 1)$. The anti-automorphism $P \mapsto \overline{P}$ is an anti-involution.

Property 1. *For all $A \in \mathbb{F}_2[XY, YX]$ one has*

$$XA = \overline{A}X.$$

Proof. It suffices to check the property for $A = (XY)^n$ and for $A = (YX)^n$, where n is a natural integer. If $A = (YX)^n$ then

$$X(YX)^n = (XY)^n X = \overline{(YX)^n} X$$

If $A = (XY)^n$, then we may assume $n \neq 0$, and

$$X(XY)^n = (YX)^{n-1} Y = (YX)^{n-1} YXX = (YX)^n X = \overline{(XY)^n} X$$

□

For $P \in \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$, define $N(P) \in \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$ by

$$N(P) = P\overline{P}.$$

Property 1 above is used to prove the following statement :

Theorem 2. *Let $P \in \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$ and write $P = A + XB$ with $A, B \in \mathbb{F}_2[XY, YX]$. Then*

$$N(P) = A\overline{A} + B\overline{B}.$$

In particular, $N(P) \in \mathbb{F}_2[XY,YX]$ and $N(P) = N(\overline{P})$.

Proof. One has

$$\begin{aligned} P\overline{P} &= (A+XB)(\overline{A+XB}) = (A+XB)(\overline{A}+XB) \\ &= A\overline{A} + AXB + XB\overline{A} + XBXB \\ &= A\overline{A} + X\overline{A}B + XB\overline{A} + \overline{B}XXB \text{ (property 1)} \\ &= A\overline{A} + X\overline{A}B + X\overline{A}B + B\overline{B} \text{ ($\mathbb{F}_2[XY,YX]$ is commutative and $X^2=1$)} \\ &= A\overline{A} + B\overline{B}. \end{aligned}$$

This proves that $N(P) = A\overline{A} + B\overline{B} \in \mathbb{F}_2[XY,YX]$. Since $\overline{P} = \overline{A} + XB$, one also has $N(\overline{P}) = N(P)$. □

The statement above suggests that N is a norm for the noncommutative ring extension of degree 2 of $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$ over $\mathbb{F}_2[XY,YX]$.

The following statement allows to decide whether or not an element P of $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$ is a right zero divisor :

Theorem 3. *Let P be an element of $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$. Then P is a right regular element if and only if $N(P) \neq 0$.*

Proof. Let P be an element of $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$. We may assume that $P \neq 0$.

1. If $N(P) = 0$ then $N(\overline{P}) = 0$ by the Theorem above. It follows that $\overline{P}P = 0$, i.e., P is not a right regular element of $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$.

2. If $N(P) \neq 0$ then, as all elements of $\mathbb{F}_2[XY,YX]$ are right regular elements of $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$, we have :

   $$\forall C \in \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1), C(P\overline{P}) = 0 \Rightarrow C = 0$$

   Thus $CP = 0 \Rightarrow (CP)\overline{P} = 0 \Rightarrow C(P\overline{P}) = 0 \Rightarrow C = 0$
   Hence P is a right regular element of $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$.

□

The preceding statement allows us to determine whether or not an element $\tau \in \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$ can serve as a transfer function for a convolutional code over the infinite dihedral group. Indeed, if $N(\tau) \neq 0$ then the multiplication-by-τ map

$$\begin{aligned} *\tau \colon \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1) &\to \mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1) \\ u &\mapsto u * \tau \end{aligned}$$

is injective, as τ is right regular.

3 Encoding

Classic convolutional codes are defined on the ring of polynomials over $\mathbb{F}_2[X]$ and have the property that the product of two monomials with a positive power of X gives a positive power of X. In our case, if we use the order given in table 1 with "positive" elements to the right of 1 and "negative" elements to the left of 1, we see that $X(XY) = Y$ and that the product of two "positive" elements can give a "negative" element.

...	YXYX	YXY	YX	Y	1	X	XY	XYX	XYXY	...

Table 1: Natural order of the monomials of $\mathbb{F}_2\{X,Y\}/(X^2-1,Y^2-1)$

Nevertheless, we can still define convolutional codes with good rate. We can see that if we choose a transfer function τ with an odd length m centered in 1 and a message u with also an odd length k, centered in 1, the convolution of u and τ add $m-1$ elements of redundancy. We choose odd lengths m and k by convention. Thus, the middle bit of a bit stream is well defined. We might also choose even lengths specifying a convention to define the middle bit of a bit stream.

We see an example in table 2 with $\tau = YX + Y + 1 + X + XY$ and $u = u_{-2}YX + u_{-1}Y + u_0 + u_1X + u_2XY$. We check that τ is a good transfer function :

$$\begin{aligned} N(\tau) &= (YX + Y + 1 + X + XY)(XY + Y + 1 + X + YX) \\ &= YXYX + YX + 1 + XY + XYXY \neq 0 \end{aligned}$$

The elements in bold type in this table are the redundancy of the code word in this example. We can see that these elements are distributed evenly on

each side of the elements on which the message is defined. Thus redundancy is placed to the left and to the right of the message and not just to the left or to the right.

τ \ u	YX	Y	1	X	XY
YX	**YXYX**	**YXY**	YX	Y	1
Y	X	1	Y	YX	**YXY**
1	YX	Y	1	X	XY
X	**XYX**	XY	X	1	Y
XY	1	X	XY	**XYX**	**XYXY**

Table 2: Redundancy (in bold type) to the left and to the right

Consequently we have the following convolution, with $G = \mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z}$ and $\forall (a,b) \in G$:

$$\begin{aligned}(u * \tau)(a,b) &= \sum_{(c,d)\in G} u(c,d)\tau((c,d)^{-1} \circ (a,b)) \\ &= \sum_{(c,0)\in G} u(c,0)\tau((-c,0) \circ (a,b)) + \sum_{(c,1)\in G} u(c,1)\tau((c,1) \circ (a,b)) \\ &= \sum_{(c,0)\in G} u(c,0)\tau(a-c,b) + \sum_{(c,1)\in G} u(c,1)\tau(c-a,b+1)\end{aligned}$$

So :

$$\begin{aligned}(u * \tau)(a,0) &= \sum_{(c,0)\in G} u(c,0)\tau(a-c,0) + \sum_{(c,1)\in G} u(c,1)\tau(c-a,1) \\ (u * \tau)(a,1) &= \sum_{(c,0)\in G} u(c,0)\tau(a-c,1) + \sum_{(c,1)\in G} u(c,1)\tau(c-a,0)\end{aligned}$$

We use the following one-to-one correspondence between G and $\mathbb{Z}$:

$$(a,b) \mapsto 2a + b$$

Thus :

$$\begin{aligned}(u * \tau)(2a) &= \sum_{c\in\mathbb{Z}} u(2c)\tau(2a-2c) + \sum_{c\in\mathbb{Z}} u(2c+1)\tau(2c-2a+1) \\ (u * \tau)(2a+1) &= \sum_{c\in\mathbb{Z}} u(2c)\tau(2a-2c+1) + \sum_{c\in\mathbb{Z}} u(2c+1)\tau(2c-2a)\end{aligned}$$

We now define a transfer function $\tilde{\tau}$ derived from the transfer function τ as follows:

$$\forall a \in \mathbb{Z},\ \tilde{\tau}(2a+1) = \tau(-2a-1)$$
$$\tilde{\tau}(2a) = \tau(2a)$$

With $\tilde{\tau}$, we have :

$$(u * \tau)(2a) = \sum_{c\in\mathbb{Z}} u(2c)\tilde{\tau}(2a-2c) + \sum_{c\in\mathbb{Z}} u(2c+1)\tilde{\tau}(2a-(2c+1))$$
$$(u * \tau)(2a+1) = \sum_{c\in\mathbb{Z}} u(2c)\tilde{\tau}(2c-(2a+1)) + \sum_{c\in\mathbb{Z}} u(2c+1)\tilde{\tau}(2c+1-(2a+1))$$

We can merge the two sums in one as follows :

$$(u * \tau)(2a) = \sum_{c\in\mathbb{Z}} u(c)\tilde{\tau}(2a-c)$$
$$(u * \tau)(2a+1) = \sum_{c\in\mathbb{Z}} u(c)\tilde{\tau}(c-(2a+1))$$

We see that the even bits of the code word $u * \tau$ will be the result of classic convolution of u and $\tilde{\tau}$, and the odd bits will be the result of convolution of u and the reciprocal of $\tilde{\tau}$. In terms of coding with a shift register, we have the circuit on figure 2.

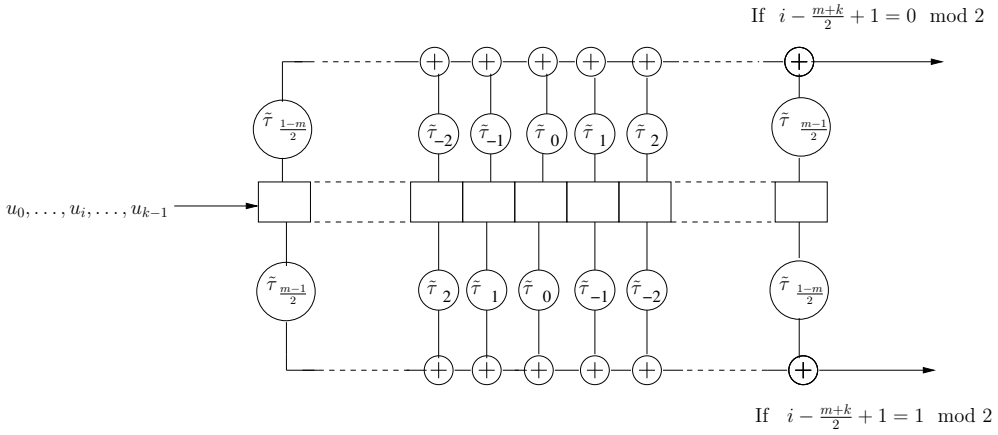

Figure 2: Circuit of encoding for the convolutional codes over $\mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z}$

We'll now see an example of encoding with $u = [1, 0, 1, 1, 1]$ and $\tau = [0, 1, 1, 0, 1]$. We check that τ can be used as transfer function :

$$N(\tau) = (Y + 1 + XY)(Y + 1 + YX) = YX + 1 + XY \neq 0$$

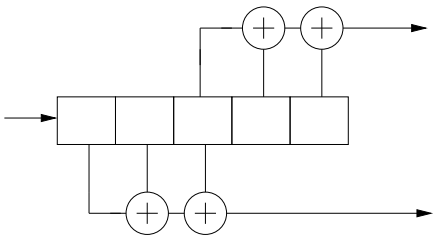

Figure 3: Example of encoding circuit with $\tau = [0, 1, 1, 0, 1]$

We have :

$$\tilde{\tau}(-2) = \tau(-2) = 0, \quad \tilde{\tau}(-1) = \tau(1) = 0$$
$$\tilde{\tau}(0) = \tau(0) = 1, \quad \tilde{\tau}(1) = \tau(-1) = 1, \quad \tilde{\tau}(2) = \tau(2) = 1$$

So $\tilde{\tau} = [0, 0, 1, 1, 1]$. And we have the encoding circuit of figure 3.

We recall that when $i - \frac{m+k}{2} + 1$ is even, we use the circuit above the shift register (corresponding to $\tilde{\tau}$) and when $i - \frac{m+k}{2} + 1$ is odd, we use the circuit below the shift register (corresponding to the reciprocal of $\tilde{\tau}$). We have the following table of the shift register with i the time, u_i the bits of the message, m_1, m_2, m_3, m_4, the register's memory (with $m_0 = u_i$) and c_i, the bit i of the code word.

i	u_i	m_1	m_2	m_3	m_4	$i - \frac{m+k}{2} + 1$	c_i
1	1	0	0	0	0	-4	0
2	0	1	0	0	0	-3	1
3	1	0	1	0	0	-2	1
4	1	1	0	1	0	-1	0
5	1	1	1	0	1	0	0
6	0	1	1	1	0	1	0
7	0	0	1	1	1	2	1
8	0	0	0	1	1	3	0
9	0	0	0	0	1	4	1

Verification :
$u = [1, 0, 1, 1, 1] \Rightarrow u(X, Y) = YX + 1 + X + XY$
$\tau = [0, 1, 1, 0, 1] \Rightarrow \tau(X, Y) = Y + 1 + XY$

$$\begin{aligned} u\tau(X, Y) &= (YX + 1 + X + XY)(Y + 1 + XY) \\ &= YXY + YX + 1 + Y + 1 + XY + XY + X + Y + X + XY + XYXY \\ &= YXY + YX + XY + XYXY \end{aligned}$$

$YXYX$	YXY	YX	Y	1	X	XY	XYX	$XYXY$
0	1	1	0	0	0	1	0	1

We find the same code word.

Our way of encoding can be put into the form of several shift registers with the same memory, where output bits are multiplexed at each time i like the circuit on figure 1.

Classic convolutional codes of rate $\frac{1}{n}$ can always be reduced to a transfer matrix of the following form[1] :

$$G = (g_0(X)\ g_1(X)\ \ldots\ g_{n-1}(X))$$

with $g_i(X) \in \mathbb{F}_2[X],\ \forall i$. The transfer function τ can be expressed as :

$$\tau(X) = g_0(X^n) + Xg_1(X^n) + \ldots + X^{n-1}g_{n-1}(X^n)$$

and the convolution is performed with $u(X^n)$.

In the case of codes on $\mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z}$, we said previously that the redundancy is added to the left and to the right, so we must use a code of minimum rate $\frac{1}{3}$. In this case, we have a transfer matrix of the following form (with n odd by convention) :

$$G = \left(g_{\frac{1-n}{2}}(X,Y)\ g_{\frac{3-n}{2}}(X,Y)\ \ldots\ g_0(X,Y)\ \ldots\ g_{\frac{n-3}{2}}(X,Y)\ g_{\frac{n-1}{2}}(X,Y)\right)$$

with $g_i(X,Y) \in \mathbb{F}_2\{X,Y\}/(X^2-1, Y^2-1),\ \forall i$. The transfer function is expressed as :

$$\tau(X,Y) = \sum_{k=\frac{1-n}{2}}^{\frac{n-1}{2}} g_k\left((XY)^{\frac{n-1}{2}}X, (YX)^{\frac{n-1}{2}}Y\right)\phi(k)$$

with ϕ the function :

$$\begin{aligned} \phi : \mathbb{Z} &\to \mathbb{F}_2\{X,Y\}/(X^2-1, Y^2-1) \\ y = 2a+b,\ b \in \mathbb{Z}/2\mathbb{Z} &\mapsto (XY)^a X^b \end{aligned}$$

The convolution is now performed with $u\left((XY)^{\frac{n-1}{2}}X, (YX)^{\frac{n-1}{2}}Y\right)$.

In order to have systematic form, it is sufficient that the central shift register repeats the bits of the message, that is to say the central index of the central register is equal to 1 and the other indexes of this register are equal to zero.

[1] Theorem 12.2 page 465 of [7]

4 Decoding

In the case of classic convolutional codes using a binary symmetric channel, the most used algorithm is the Viterbi algorithm [9]. This algorithm uses a trellis whose edges connect each state of memory to the next state according to the bit entering the shift register at each time. Each edge has a label with the output of the circuit at each time according to the memory associated with each edge.

In the case of convolutional codes over the infinite dihedral group, we have two trellis, one related to the circuit above the shift register, the other related to the circuit below the shift register. Since the two circuits alternate in encoding, the two trellis alternate in decoding (figure 4). So we use an adapted Viterbi algorithm to decode these codes, using two trellis instead of one.

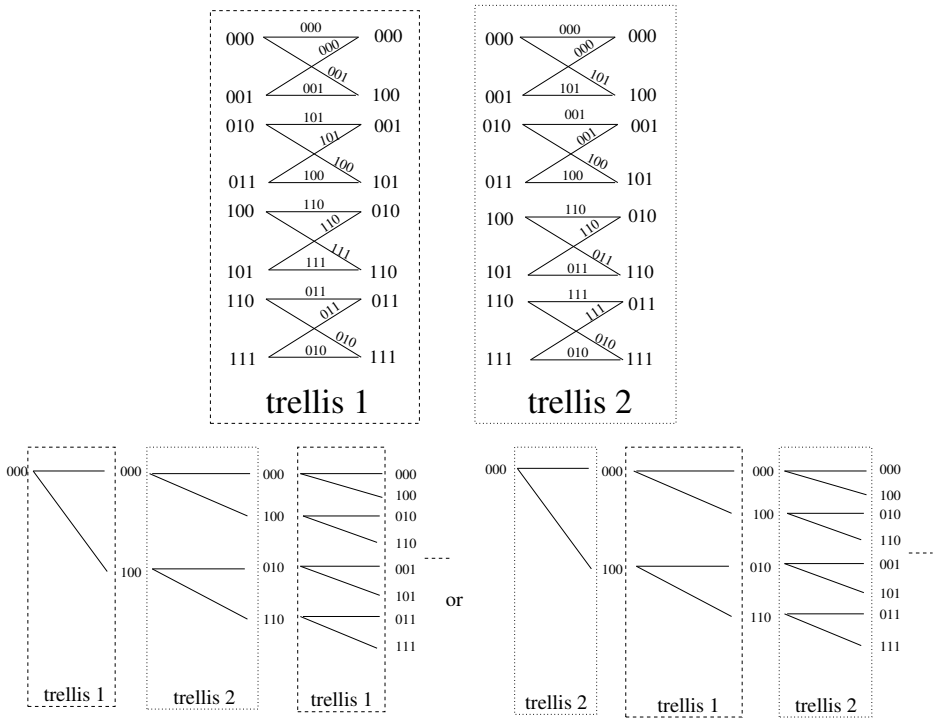

Figure 4: Example of adapted Viterbi algorithm

5 Properties of these codes

Firstly, the convolutional codes over the infinite dihedral group have a memory of length m (with m the length of the transfer function in binary form) like the classic convolutional codes. We have two different outputs of the shift register but they are alternating, so there is the same number of operations as we have only one output circuit. Thus the encoding complexity is the same as for classic convolutional codes. For decoding, we have two trellis that alternate, so we also have the same decoding complexity as for classic convolutional codes.

Secondly, to determine the free distance of these codes and to compare it with the free distance of classic convolutional codes, we have realized some tests whose results are in table 3. We see that, for the same memory length and the same rate, we have the same maximum free distance for the codes over the infinite dihedral group as for the classic codes.
We know that the free distance of a code depends on the trellis of the code. In the case of codes over $\mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z}$, two trellis alternate in decoding. If we start with trellis 1 then we get a free distance that may differ from the one we get if we start with trellis 2. Therefore, the free distance depends on the trellis we start with. Since the trellis we start with, depends on the length of the code word, we can choose a starting trellis by adapting the length of the message. If we make this choice, we see that the number of transfer matrices that achieve the maximum free distance $\left(G^{(\max d_f)}_{\mathbb{Z}\rtimes\mathbb{Z}/2\mathbb{Z}}\right)$ is greater than the number of transfer matrices that achieve the maximum free distance in the case of classic convolutional codes $\left(G^{(\max d_f)}_{\mathbb{Z}}\right)$.
If we don't adapt the message's length, then the free distance is the minimum of the two free distances associated with each trellis. In this case, we have the same number of transfer matrices that achieve the maximum free distance of codes over $\mathbb{Z} \rtimes \mathbb{Z}/2\mathbb{Z}$ as over $\mathbb{Z}$.

Finally, we compare the difficulty for an eavesdropper to recognize the code in non-cooperative context, between convolutional codes on D_∞ and classic convolutional codes. In classic convolutional codes, we find n by use a method called rank criterion [6]. Here we have the same criterion because the n output bits at time i are dependant. However, we can not use exactly the same algorithm to recover the parity check matrix. Indeed, the p^{th} blocks of n bits, $p = 0 \mod 2$, belong to a classic code C and the other blocks belong to the reciprocal of C. So the attacker separates the bit stream into two different streams. Then, the attacker applies the algorithm of code recognition to the first stream and finds the parity check matrix and

m	rate	systematic form	$\max(d_f)$	number of $G_{\mathbb{Z}}^{(\max(d_f))}$	number of $G_{\mathbb{Z}\rtimes\mathbb{Z}/2\mathbb{Z}}^{(\max(d_f))}$	number of $G_{\mathbb{Z}\rtimes\mathbb{Z}/2\mathbb{Z}}^{(\max(d_f))}$ choice of starting trellis
3	1/3	yes	6	7	3	7
3	1/3	no	8	3	3	7
3	1/5	yes	12	4	4	12
3	1/5	no	13	10	10	70
5	1/3	yes	9	6	6	32
5	1/3	no	12	6	6	144
3	1/7	yes	17	15	15	105
3	1/7	no	18	56	56	656
5	1/5	yes	17	90	90	1636

Table 3: Maximum free distance and number of transfer matrix achieving this maximum for code of memory length and rate given

then the generator matrix. The other code is nothing but the reciprocal of the code he recognized. So the difficulty seems to be the same.

6 Conclusion

These codes over the infinite dihedral group are well defined mathematically and electronically and have the same complexity as classic convolutional codes. They have the same free distance, but we obtain more optimal codes. Moreover, the highest amount of optimal codes can be used to design non-generic dedicated proprietary transmitter/receiver. Indeed, the greater is the choice of optimal encoder with same parameters and performance to encode information, the more it is possible to implement dedicated encoder/decoder which can not be decoded by those of its competitors. However, they are not more resistant to code recognition because the group is too close to the group of integers and the encoding is not time-varying. In future work, we will study convolutional block codes over finite noncommutative groups, where the encoding varies at each message.

References

[1] IEEE International Symposium on Information Theory, ISIT 2009, June 28 - July 3, 2009, Seoul, Korea, Proceedings. IEEE (2009)

[2] Adkins, W.A., Weintraub, S.H.: Algebra an approach via module theory. Graduate Texts in Mathematics. Springer, New York, berlin, heidelberg (1992)

[3] Cote, M., Sendrier, N.: Reconstruction of convolutional codes from noisy observation. In: ISIT [1], pp. 546–550

[4] Elias, P.: Coding for Two Noisy Channels. In: Information Theory, The 3rd London Symposium, pp. 61–76. Buttersworth's Scientific Publications (1955)

[5] Forney G.D., J.: The viterbi algorithm. Proceedings of the IEEE **61**(3), 268–278 (1973)

[6] Marazin, M., Gautier, R., Burel, G.: Blind recovery of k/n rate convolutional encoders in a noisy environment. EURASIP J. Wireless Comm. and Networking **2011**, 168 (2011)

[7] Moon, T.K.: Convolutional codes. In: Error Correction Coding: Mathematical Methods and Algorithms, chap. 12, pp. 452–580. Wiley-Interscience (2005)

[8] Neubauer, A.: Convolutional codes. In: Coding Theory: Algorithms, Architectures and Applications, chap. 3, pp. 112–177. Wiley-Interscience (2007)

[9] Viterbi, A.: Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE Transactions on Information Theory **13**(2), 260–269 (1967)