SahuarUS
Revista electrónica de Matemáticas
Vol.7 / No.2 / Diciembre 2023, pp. 14-28
SahuarUS
REVISTAS UNISON
"El saber de mis hijos hará mi grandeza"

Códigos cíclicos para desarrollar algoritmos de ADN

Nayeli Adriana González Martínez[1] y Adolfo Maceda Méndez[2]
Universidad Tecnológica de la Mixteca.
[1]gonaye06@hotmail.com,[2]admm@mixteco.utm.mx

Resumen

La computación del ADN es una forma de computación que usa moléculas de ADN y moléculas biológicas en lugar de las tecnologías informáticas tradicionales basadas en silicio, una de las razones por las que empezó a ser interesante estudiar la computación del ADN es que un sólo gramo de ADN con un volumen de 1 cm^3 *puede contener tanta información como un billón de discos compactos, aproximadamente* 750 *terabytes. Otra característica importante de los procesos del ADN es la capacidad de realizar muchas operaciones en paralelo, lo que es posible gracias a la capacidad de replicación del ADN. Richard Feynman introdujo por primera vez el cálculo molecular a principios de 1960, pero fue hasta 1994 cuando Leonard Adleman, de la Universidad del Sur de California, demostró un uso del ADN para resolver el problema Hamiltoniano de* 7 *puntos. Después de su trabajo, diversos investigadores han encontrado nuevas aplicaciones para la computación del ADN, y en diversas áreas se ha descubierto la importancia de llevar los cálculos a nivel molecular. El objetivo del artículo es mostrar cómo se utiliza la teoría de campos finitos para construir códigos cíclicos que permiten modelar la estructura del ADN. Esta modelación es importante porque se conocen algoritmos basados en ADN que permiten resolver, mediante procesos químicos, algunos problemas clásicos como el del árbol de expansión mínima y el del coloreado de una gráfica con tres colores. Actualmente, se están desarrollando simulaciones de estos algoritmos mediante computadoras.*

Palabras Clave: Códigos aditivos, Códigos lineales, Computación del ADN.

DOI: 10.36788/sah.v7i2.141
Recibido: 21 de junio del 2023.
Aceptado: 27 de noviembre del 2023.

1. Introducción

El ADN es una molécula de gran tamaño que almacena y transmite toda la información necesaria para el desarrollo de todas las funciones biológicas necesarias para un organismo, está formado por una unión paralela de dos cadenas, cada cadena formada por cuatro diferentes nucleótidos: Adenina, Guanina, Timina y Citosina. Su estructura fue dilucidada en 1953 por los científicos Watson y Crick, descubrimiento que años más tarde los haría ganadores del premio Nobel [8].

ISSN 2448 5365
Universidad de Sonora
Los contenidos de este artículo están bajo una licencia de Creative Commons Atribución No Comercial - Sin Obra Derivada 4.0 Internacional

Una hebra simple de ADN es una sucesión de los cuatro nucleótidos mencionados, cuyos extremos son químicamente polares y son llamados extremos $5'$ y $3'$. El complemento Watson-Crick de una hebra de ADN se obtiene al reemplazar Adenina por Timina y viceversa; cada Guanina con Citosina y viceversa; e intercambiar los extremos $3'$ y $5'$. Una hebra simple se une a su complemento Watson-Crick por medio de un proceso denominado hibridación específica, formando así la estructura conocida como doble hélice de ADN.

La computación del ADN ofrece grandes ventajas a los investigadores debido a la gran capacidad de almacenamiento de información, y a la realización de procesos en paralelo, sin embargo, su potencial se limita precisamente por la composición química del ADN. Varios autores han propuesto diferentes técnicas para construir un conjunto de palabras código de ADN que es poco probable que formen enlaces indeseables entre sí por hibridación.

El objetivo de este trabajo es usar la teoría de códigos cíclicos para construir códigos que son adecuados para la computación del ADN, en particular construimos códigos lineales y aditivos sobre $GF(4)$, que son adecuados para esta aplicación, basándonos en el artículo de Abualrub [1].

Este trabajo está estructurado de la siguiente manera: En la sección 2 se explica brevemente qué es la computación del ADN y el proceso de hibridación de las hebras de ADN para enteder por qué el interés de construir estas codificaciones, en la sección 3 se exponen algunos conceptos de teoría de códigos que son básicos para entender cómo son los códigos que permitirían implementar los algoritmos de ADN. En la sección 4 se construyen códigos usando campos finitos. En la sección 5, se construyen códigos adecuados para la computación del ADN y finalmente en la sección 6 se construyen códigos aditivos de longitud 7 que mejoran los algoritmos de ADN.

2. La computación del ADN

La computación del ADN inició en 1994 cuando Leonard Adleman resolvió el problema de encontrar un camino Hamiltoniano de 7 puntos dada un gráfica mediante manipulaciones de moléculas de ADN en un tubo de ensayo. Adleman encontró una ruta que pasa por todos los vértices de la gráfica exactamente una vez. Este problema debido a su complejidad pertenece a los denominados problemas NP-completos, los cuales son muy difíciles de resolver, pues no se pueden resolver en tiempo polinomial.

Después del trabajo de Adleman, diversos investigadores han utilizado la computación del ADN como método para resolver problemas complejos de diferentes áreas. Por ejemplo, las codificaciones de ADN han sido utilizadas para resolver diversos problemas como:

- El árbol de expansión mínima ([4]).
- La ruta más corta en una gráfica ([3]).
- El coloreado de una gráfica con tres colores ([3]).
- El problema del conjunto dominante ([3]).

Sin embargo, para poder realizar estos algoritmos mediante la computación del ADN es necesario contar con una codificación del ADN que sea útil y que dé como resultado el mínimo de enlaces indeseables posibles.

La computación del ADN ofrece ciertas ventajas, como la gran capacidad de almacenamiento de las moléculas de ADN, pues un gramo de ADN puede contener alrededor de 750 terabytes. Además, algo que ha llamado la atención de los investigadores es la capacidad del ADN de realizar muchos procesos en paralelo, esto se debe a la capacidad de replicación del ADN, y constituye una importante ventaja pues disminuye el tiempo de respuesta, al hacer simultáneamente varios procesos.

El potencial de la computación del ADN se ve limitado por su estructura química, debido a que el ADN es una doble hélice, formada por una hebra simple de ADN y su complemento Watson-Crick, unidos mediante hibridación específica, pues al formarse la doble hélice pueden ocurrir errores y dar como resultado cadenas de ADN que no funcionan, es decir, se obtienen falsos positivos o falsos negativos. Los falsos positivos resultan cuando se da la hibridación no específica, es decir, se une una hebra de ADN con el complemento Watson-Crick de una hebra diferente. La hibridación no específica también ocurre cuando se une una hebra de ADN con el reverso de una hebra distinta. Un falso negativo ocurre cuando no se da la hibridación entre una hebra y su complemento Watson-Crick.

Diversos autores han propuesto diferentes técnicas para construir conjuntos de cadenas de ADN en los cuales sea poco probable que se den enlaces indeseables por hibridación [2],[7]. En este trabajo se construyen códigos aditivos sobre $GF(4)$. La principal ventaja de utilizar códigos aditivos sobre códigos lineales es que para la misma longitud de código hay más códigos aditivos que lineales, además, es más probable encontrar códigos aditivos que tengan más palabras código que un código lineal con la misma distancia de Hamming [1].

Ejemplo 2.1 *Consideremos códigos de longitud* 3 *y distancia* 2 *sobre el campo finito* $\mathbb{Z}_2$*:*

- $C_1 = \{000, 011, 101, 110\}$
- $C_2 = \{011, 101, 110\}$
- $C_3 = \{000, 101, 110\}$
- $C_4 = \{000, 011, 110\}$
- $C_5 = \{000, 011, 101\}$
- $C_6 = \{000, 011\}$
- $C_7 = \{000, 101\}$
- $C_8 = \{000, 110\}$
- $C_9 = \{011, 101\}$
- $C_{10} = \{011, 110\}$

- $C_{11} = \{101, 110\}$

De los cuales C_1, C_6, C_7 y C_8 son lineales y C_1, C_2, C_6, C_7, y C_8 son aditivos. Es decir, hay más códigos aditivos que lineales, y encontramos que C_2, el cual es aditivo, tiene más palabras código que C_6, C_7 y C_8.

3. Conceptos principales de teoría de códigos

La teoría de códigos es una especialidad matemática que trata de las leyes de la codificación de la información. A grandes rasgos, codificar es transformar una información en una señal convenida para su comunicación. Decodificar sería el proceso inverso y complementario del anterior por el cual la señal comunicada es transformada en la información original [10].

Para poder entender un poco acerca de los códigos de los cuales trataremos en este trabajo, necesitamos las siguientes definiciones:

Definición 3.1 *Un alfabeto es un conjunto finito $\mathcal{A} = \{a_1, a_2, \ldots, a_q\}$ cuyos elementos son llamados símbolos y q (el número de elementos de $\mathcal{A}$) es llamado la raíz de $\mathcal{A}$.*

A partir de los elementos de un alfabeto, se tiene lo siguiente:

Definición 3.2 *Una palabra de longitud n sobre $\mathcal{A}$ es una sucesión de n elementos de $\mathcal{A}$; en general, escribiremos las palabras de la siguiente forma:*

$$a = a_{i1}a_{i2}\ldots a_{in}, \qquad a_{ik} \in \mathcal{A}, k = 1, 2, 3, \ldots, n.$$

Denotaremos por $\mathcal{A}^n$ a todas las palabras de longitud n y por $\mathcal{A}^*$, al conjunto que contiene a todas las palabras, es decir,

$$\mathcal{A}^* = \bigcup_{n\in\mathbb{N}} \mathcal{A}^n.$$

Definición 3.3 *Si $\mathcal{A} = \{a_1, a_2, \ldots, a_q\}$ es un alfabeto, un código q-ario sobre $\mathcal{A}$ es un subconjunto C de $\mathcal{A}^*$. Los elementos de C son llamadas palabras código. El número $M = |C|$, el cardinal de C, es llamado tamaño del código.*

Se tienen dos tipos de códigos según la longitud de sus palabras,

- Códigos de bloque: La longitud de todas las palabras es la misma, y se dice que tiene parámetros (n, M), donde n es la longitud de las palabras y M es el tamaño del código.
- Códigos de longitud variable: Está conformado por palabras de diferente longitud.

En este trabajo sólo utilizaremos palabras de longitud fija, por lo que siempre que nos refiramos a un código, se asumirá que se trata de un código de bloque.

Consideremos una secuencia de información $u_0u_1u_2u_3\ldots$ de símbolos de información u_i tomados de un alfabeto $\mathcal{A}$. Si agrupamos esta secuencia en bloques de longitud k de la siguiente manera:

$$\underbrace{u_0u_1\ldots u_{k-1}}\underbrace{u_ku_{k+1}\ldots u_{2k-1}}\underbrace{u_{2k}u_{2k+1}\ldots u_{3k-1}},$$

cada uno de estos bloques de información forman una palabra, la que llamaremos *palabra de información de longitud k*. Como cada símbolo de una palabra de información puede tomar q valores, se tiene que *el número de palabras código* es $M = q^k$.

Existe un parámetro en los códigos que nos permite saber que tan cerca está una palabra de otra, y es útil para poder detectar y corregir errores dentro de un código.

Definición 3.4 *La distancia de Hamming entre dos palabras código* $\mathbf{b} = b_0b_1\ldots b_{n-1}$ *y* $\mathbf{b}' = b'_0b'_1\ldots b'_{n-1}$*, denotada por* $H(\mathbf{b},\mathbf{b}')$*, se define como:*

$$H(\mathbf{b},\mathbf{b}') = |\{0 \leq i < n : b_i \neq b'_i\}|.$$

Es fácil ver que la Distancia de Hamming es una métrica en el conjunto $\mathcal{A}^n$, (Vea [6]).

Definición 3.5 *Dado un código C, se define la distancia de C, y se denota por* $d(C)$*, como la menor distancia no nula entre sus palabras código, es decir,*

$$d(C) = mín\{H(x,y) : x,y \in C,\ x \neq y\}.$$

Esta definición es importante pues ayuda a saber cuántos errores puede detectar y corregir un código, mediante el siguiente resultado.

Teorema 3.6 *Un código C con distancia* $d(C)$ *puede detectar* $d(C)-1$ *errores, y puede corregir* $\lfloor\frac{d(C)-1}{2}\rfloor$.

Sin embargo, no ahondaremos en este tema en el desarrollo de este trabajo, el lector interesado en este tema puede consultar [6].

4. Construcción de códigos usando campos finitos

Para codificar y decodificar de manera más práctica y eficiente, es útil dotar al alfabeto $\mathcal{A}$ de una estructura algebraica. Fijamos al alfabeto $\mathcal{A} = \mathbb{F}_q$, el campo finito de q elementos. El conjunto de cadenas de longitud n, es decir el conjunto $\mathcal{A}^n$ es un espacio vectorial sobre $\mathbb{F}_q$, de dimensión n, el cual puede ser identificado como:

$$\mathbb{F}_q^n = \{(x_0,x_1,\ldots x_{n-1}) : x_i \in \mathbb{F}_q, 0 \leq i \leq n-1\},$$

mediante la asignación $x_0x_1\ldots x_{n-1} \mapsto (x_0,x_1,\ldots,x_{n-1})$, con la cual definimos la suma de las palabras $a_0a_1\ldots a_{n-1}$ y $b_0b_1\ldots b_{n-1}$ como

$$a_0a_1\ldots a_{n-1} + b_0b_1\ldots b_{n-1} = (a_0,a_1,\ldots,a_{n-1}) + (b_0,b_1,\ldots,b_{n-1}),$$

y la multiplicación por escalar $\alpha(a_0a_1\ldots a_{n-1})$, con $\alpha \in \mathbb{F}_q$, como

$$\alpha(a_0a_1\ldots a_{n-1}) = \alpha(a_0,a_1,\ldots,a_{n-1}).$$

Con esto podemos definir los códigos aditivos y los códigos cíclicos como sigue:

Definición 4.1 *Un código aditivo C sobre el campo finito $\mathbb{F}_q$, es un subgrupo aditivo de $\mathbb{F}_q^n$. Es decir, para cada par de palabras código $a_0a_1\dots a_{n-1}$, y $b_0b_1\dots b_{n-1} \in C$, su suma es una palabra código de C.*

Definición 4.2 *Un código C es lineal si es un subespacio de $\mathbb{F}_q^n$. Un código lineal de longitud n, dimensión k, y distancia d es llamado un $[n,k,d]$ código.*

Ejemplo 4.3 *Sea $C = \{011, 101, 000, 110\}$. Considerado como un código sobre $\mathbb{Z}_3$, este código no es lineal, puesto que $2(011) = 022$ no está en C, sin embargo, visto como un código sobre $\mathbb{Z}_2$, sí es aditivo.*

En este trabajo serán de suma importancia los códigos cíclicos, pues se espera que mejoren los algoritmos de ADN, y cuya definición es la siguiente:

Definición 4.4 *Un código C es cíclico si siempre que la palabra $c_0c_1\dots c_{n-1} \in C$, entonces $c_{n-1}c_0\dots c_{n-2} \in C$.*

Ejemplo 4.5 *Sea $C_1 = \{011, 101, 110\}$ un código sobre el campo $\mathbb{Z}_2$, el cual no es un código aditivo ni lineal, puesto que 000 no está en C_1, sin embargo, sí es un código cíclico. Y por otro lado, el código $C_2 = \{000000, 000111, 111000, 111111\}$ sobre $\mathbb{Z}_2$ es un código lineal pero no es cíclico.*

Para poder explotar muchos resultados que se tienen en álgebra, haremos una relación entre un anillo de polinomios y los códigos cíclicos, para ello, notemos lo siguiente.

Nota. *Sea $\mathbb{F}$ un campo, consideremos el anillo $\mathbb{F}[x]/\langle x^n - 1\rangle$. A este anillo lo denotaremos como $\mathbb{F}^{(n)}[x]$, y a sus elementos los denotaremos simplemente $c(x)$ en lugar de $[c(x)]$.*

Teorema 4.6 *Si $\mathbb{K}$ es un campo e I el ideal pricipal de $\mathbb{K}[x]$ generado por el polinomio $p(x) = a_0 + a_1x + \cdots a_nx^n$ con $a_n \neq 0$, entonces para cada $[r(x)] \in \mathbb{K}[x]/I$, existe un único polinomio $q(x) = b_0 + b_1x + \cdots b_{n-1}x^{n-1}$ tal que $[r(x)] = [q(x)]$.*

Puede consultar la demostración de este resultado en [5].

Consideremos un elemento $c(x) \in \mathbb{F}^{(n)}[x]$, por el teorema anterior, todo elemento de $\mathbb{F}^{(n)}[x]$ puede ser representado únicamente por un polinomio de grado $n-1$, por lo tanto, podemos suponer que $c(x)$ es de grado $n-1$, al multiplicarlo por x, obetenemos:

$$xc(x) = c_0x + c_1x^2 + \cdots c_{n-2}x^{n-1} + c_{n-1}x^n,$$

es decir,

$$xc(x) = c_{n-1} + c_0x + \cdots + c_{n-2}x^{n-1},$$

puesto que $x^n = 1$ en $\mathbb{F}^{(n)}[x]$, esto genera lo que se llama **desplazamiento cíclico**.

El anillo $\mathbb{F}^{(n)}[x]$ es también un espacio vectorial sobre $\mathbb{F}$, y la multiplicación por escalar es como sigue:

Si $a \in \mathbb{F}$ y $c(x) = c_0 + c_1x + \cdots + c_{n-1}x^{n-1} \in \mathbb{F}^{(n)}[x]$, entonces,

$$ac(x) = ac_0 + ac_1x + \cdots + ac_{n-1}x^{n-1}.$$

Ahora construiremos códigos cíclicos lineales a partir de un ideal en el anillo de polinomios, de la siguiente forma:

Los códigos cíclicos lineales pueden definirse a partir de un ideal, para lo cual necesitamos establecer una correspondencia entre los códigos sobre un campo finito y el anillo de polinomios. Consideremos las siguientes funciones:

$$\phi : \mathbb{F}_q^n \to \mathbb{F}[x],$$

dada por

$$\phi(c_0, \ldots, c_{n-1}) = c_0 + c_1x + \cdots + c_{n-1}x^{n-1},$$

y además,

$$\pi : \mathbb{F}[x] \to \mathbb{F}^{(n)}[x],$$

dada por

$$\pi(p(x)) = [p(x)].$$

Teorema 4.7 *La función $\pi \circ \phi$ es una transformación lineal. Si C es un código cíclico y lineal, $\pi \circ \phi(C)$ es un ideal. Más aún, $\pi \circ \phi\big|_C : C \to (\pi \circ \phi)(C)$ es un isomorfismo de espacios vectoriales.*

Por este resultado, podemos trabajar indistintamente con elementos de C o con la correspondiente clase de equivalencia $c(x)$.

Ahora veamos la relación que existe entre los ideales de $\mathbb{F}^{(n)}[x]$ y los códigos cíclicos.

Teorema 4.8 *Sea J un ideal no cero de $\mathbb{F}^{(n)}[x]$. El conjunto*

$$\{(a_0, \ldots, a_{n-1}) \in \mathbb{F}^n : [a_0 + a_1x + \cdots + a_{n-1}x^{n-1}] \in J\},$$

es un código cíclico.

Y además,

Teorema 4.9 *Sea J un ideal no cero en $\mathbb{F}^{(n)}[x]$. Existe un único polinomio mónico $g(x) \in \mathbb{F}[x]$ de grado r tal que,*

1. *$J = \langle g(x) \rangle / \langle x^n - 1 \rangle$.*
2. *$g(x)$ divide a $x^n - 1$.*

Además, si C es el código cíclico lineal correspondiente a J, entonces la dimensión de C es $n - r$.

Vea las demostraciones en [5].

5. Construcción de códigos para la computación del ADN

En esta sección vamos a construir el campo $GF(4)$, el cual es un campo de Galois o campo finito de 4 elementos. Para ello, tomaremos como base el campo de los números binarios $(GF(2), +, *)$, donde $GF(2) = \{0, 1\}$ es el anillo de los enteros módulo 2, con las operaciones suma " $+$ " y producto " $*$ " usuales en este anillo. Las pruebas de los teoremas que se presentan pueden ser revisadas por los lectores en [5].

Sea $GF(2)[x]$, el anillo de los polinomios en una variable con coeficientes en $GF(2)$. Tomemos el polinomio $p(x) = x^2 + x + 1 \in GF(2)[x]$, el cual sabemos que es irreducible en $GF(2)$, ya que es de grado 2 y que no tiene raíces en el campo. Ahora consideremos el anillo cociente $GF(2)[x]/\langle p(x)\rangle$, que resulta ser un campo, donde $\langle p(x)\rangle$ es el ideal generado por el polinomio $p(x)$. En este caso,

$$GF(2)/\langle p(x)\rangle = \{a(x) + \langle p(x)\rangle : a(x) \in GF(2)[x]\}.$$

Como $GF(2)[x]$ es un anillo euclidiano, usando el algoritmo de la división, se tiene que para cualquier elemento $a(x) \in GF(2)[x]$,

$$a(x) = p(x)q(x) + r(x), \text{ con } 0 \leq grad(r(x)) \leq 1.$$

Por lo tanto, un representante de la clase de $a(x)$, dada por $[a(x)] = a(x) + \langle p(x)\rangle$ es $a_0 + a_1x$, con $a_0,\ a_1 \in GF(2)$.

Por consiguiente, los elementos del anillo $GF(2)/\langle p(x)\rangle$ se pueden identificar mediante el conjunto $GF(4)$, además, por el hecho de ser $p(x)$ irreducible, $\langle p(x)\rangle$ es maximal, y por lo tanto $GF(4)$ es un campo.

$$GF(4) = \{a_0 + a_1x : a_0, a_1 \in GF(2)\} = \{0, 1, \omega, 1 + \omega\} = \{0, 1, \omega, \bar{\omega}\},$$

donde $\omega = [x] = x + \langle p(x)\rangle$, y $\bar{\omega} = 1 + \omega$, además $\bar{\omega} = \omega^2$, y $1 + \omega + \omega^2 = 0$.

5.1. Códigos asociados con el ADN

Las cadenas de ADN tienen características muy especiales que resultan importantes para el desarrollo de algoritmos, una de estas características es que cada hebra de ADN se asocia a una hebra complementaria, su complemento Watson-Crick.

Para poder codificar las hebras de ADN y usar la teoría de códigos para su estudio, asociaremos códigos sobre el conjunto $\{A, C, G, T\}$ con códigos sobre $GF(4) = \{0, 1, \omega, \bar{\omega}\}$, es decir, asociaremos 0 con Adenina (A), ω con Citosina(C), $\bar{\omega}$ con Guanina(G) y 1 con Timina(T). Para definir el complemento de una palabra código en $GF(4)$, definiremos primero el complemento de un elemento de $GF(4)$.

Definición 5.1 *Sea $a \in GF(4)$, el complemento de a, denotado por a^c, se define como $a^c = a + 1$.*

Con esto, definamos:

Definición 5.2 *El complemento de una palabra* $\mathbf{u} = u_0 \dots u_{n-1} \in GF(4)$ *es* $\mathbf{u}^c = u_0^c \dots u_{n-1}^c$, *y el reverso de* $\mathbf{u}$ *es* $\mathbf{u}^r = u_{n-1} \dots u_0$.

Note que si se da una asociación diferente a la propuesta para el conjunto de nucleótidos que forman el ADN con el campo $GF(4)$, no se tienen las definiciones anteriores, puesto que con esta asociación se tiene que el complemento de A es T, y el de C es G.

Ahora definamos códigos que tienen características particulares que resultan importantes para los códigos de ADN.

Definición 5.3 *Un código aditivo* C *de longitud* n *sobre* $GF(4)$ *es llamado reversible si* $u^r \in C$ *para todo* $u \in C$.

Ejemplo 5.4 *El código* $C = \{110, 011, 101, 000\}$ *es reversible, pero* $C' = \{111, 100, 011, 000\}$ *no lo es.*

El reverso de una palabra código tiene su equivalente en el anillo de polinomios, esto es lo que se llama *recíproco de un polinomio.*

Definición 5.5 *Para cada polinomio* $p(x) = p_0 + p_1 x + \cdots + p_r x^r$ *con* $p_r \neq 0$, *definimos el recíproco de* $p(x)$ *como el polinomio* $p^*(x) = x^r p(1/x) = p_r + p_{r-1}x + \cdots + p_0 x^r$. *Además diremos que un polinomio es auto-recíproco si* $p(x) = p^*(x)$.

Ejemplo 5.6 *Consideremos el polinomio* $p(x) = 1 + x + x^2 + x^3 + x^4 + x^5 + x^6$, *notemos que* $p^*(x) = x^6 p(1/x) = x^6(1 + \frac{1}{x} + \frac{1}{x^2} + \frac{1}{x^3} + \frac{1}{x^4} + \frac{1}{x^5} + \frac{1}{x^6}) = 1 + x + x^2 + x^3 + x^4 + x^5 + x^6 = p(x)$. *Por lo tanto* $p(x)$ *es auto-recíproco.*

Definición 5.7 *Un código aditivo* C *sobre* $GF(4)$ *es llamado complemento si* $u^c \in C$ *para todo* $u \in C$.

Definición 5.8 *Un código aditivo* C *sobre* $GF(4)$ *es llamado complemento cíclico reversible si,*

1. C *es cíclico.*
2. C *es reversible.*
3. C *es complemento.*

Recordemos ahora un resultado muy importante en teoría de anillos, llamado *Teorema de Correspondencia.*

Teorema 5.9 ***Teorema de Correspondencia****. Sea* A *un anillo e* I *un ideal de* A. *Sea* $\varphi : A \to A/I$ *la proyección canónica, dada por* $\varphi(a) = a + I$. *Existe una correspondencia biunívoca entre ideales de* A/I *e ideales de* A *que contienen a* I.

En adelante, consideraremos a los elementos de los códigos cíclicos como polinomios, puesto que ya vimos que hay una correspondencia entre ellos y los ideales en $\mathbb{F}^{(n)}[x]$. Así que estudiaremos algunos resultados que sólo se satisfacen en el campo $GF(4)$.

Definición 5.10 *Sea C un código cíclico lineal correspondiente a $J = \langle g(x)\rangle/\langle x^n - 1\rangle$, un ideal de $GF(4)^{(n)}[x]$, a su vez por el Teorema de Correspondencia a J le corresponde un ideal J' de $GF(4)^{(n)}[x]$, tal que $J = J'/\langle x^n - 1\rangle$. Como cada elemento de J es una clase de equivalencia, definimos el grado de $[f(x)] \in J$ como el grado del polinomio $f(x) \in J'$.*

Así, el grado de $f(x) \in C$ será el grado de $[f(x)] \in J$, y lo denotaremos como $grad(f(x))$.

El siguiente teorema muestra un resultado importante sobre el campo $GF(4)$.

Teorema 5.11 *Sea C un $[n, k, d]$-código cíclico lineal de longitud n sobre $GF(4)$. Entonces $C = \langle g(x)\rangle$, donde $g(x)$ es un polinomio mónico de grado r que divide a $x^n - 1$ sobre $GF(4)$, y $k = n - r$.*

Es importante ahora ver cómo son los generadores de un código cíclico aditivo, para ello, consideremos la función *traza*,

$$Tr : GF(4) \to GF(2), \tag{1}$$

definida por $Tr(x) = x + x^2$.

En particular, $Tr(0) = Tr(1) = 0$, $Tr(\omega) = \omega + \omega^2 = 1$, puesto que $1 + \omega + \omega^2 = 0$, $\bar{\omega} = \omega + 1$ y $Tr(\bar{\omega}) = 1$.

Es fácil ver que la función traza es un homomorfismo de grupos aditivos.

El siguiente teorema nos dará una forma de describir un tipo particular de códigos cíclicos aditivos mediante sus polinomios generadores.

Teorema 5.12 *Sea C un $(n, 2^k)$-código cíclico aditivo de longitud n sobre $GF(4)$. Entonces $C = \langle \omega p(x) + q(x)\rangle$, donde $p(x)$ y $r(x)$ son polinomios binarios que dividen a $x^n - 1$ $(mod\ 2)$, $r(x)$ divide a $\frac{q(x)(x^n-1)}{p(x)} (mod\ 2)$, y $k = 2n - grad(p(x)) - grad(r(x))$.*

Observaciones 5.1 *Cuando algún polinomio generador es nulo, se tiene lo siguiente.*

- *Si $p(x) = 0$, entonces $k = n - grad(r(x))$, y si $r(x) = 0$, se tiene que $k = n - grad(p(x))$.*
- *Como para cualquier subgrupo se tiene que $\langle a, b\rangle = \langle a, b - q\rangle$, nosotros asumiremos que $grad(q(x)) < grad(p(x))$.*

Ahora veamos algunos resultados sobre códigos cíclicos reversibles y los polinomios generadores.

Teorema 5.13 *Un código lineal C cíclico $\langle g(x)\rangle$ sobre $GF(4)$ es reversible si y sólo si $g(x)$ es auto-recíproco.*

Demostración. Supongamos primero que C es reversible, para ello consideremos el conjunto de polinomios recíprocos de los polinomios de C, $C' = \{f^*(x) : f(x) \in C\}$, veamos que $C' = \langle g^*(x)\rangle$, entonces, tomemos $f^*(x) \in C'$.

$$\begin{aligned} f^*(x) &= x^{n-1}f(x^{-1}) \\ &= x^{n-1}g(x^{-1})j(x^{-1}) \\ &= (x^{n-r-1}j(x^{-1}))(x^r g(x^{-1})) \\ &= j^*(x)g^*(x). \end{aligned}$$

Por lo tanto, $f^*(x) \in \langle g^*(x)\rangle$. La otra inclusión es clara. Así, $C' = \langle g^*(x)\rangle$. Además, como C es reversible, entonces $\langle g(x)\rangle = C = C' = \langle g^*(x)\rangle$.

Así $g^*(x)$ también genera a C y al ser $g(x)$ el polinomio de grado mínimo que genera a C, se tiene que $g(x) = g^*(x)$, es decir, $g(x)$ es auto-recíproco.

Recíprocamente, supongamos que $g(x) = g^*(x) = x^r g(\frac{1}{x})$, donde $r = grad(g(x))$. Tomemos $p(x) \in C$, con $grad(p(x)) = m > r$. Debemos mostrar que $p^*(x) \in C$, es decir, $p^*(x)$ es un múltiplo de $g(x)$. Calculemos $p^*(x)$,

$$\begin{aligned} p^*(x) &= x^m p(\frac{1}{x}) = x^m k(\frac{1}{x})g(\frac{1}{x}) \\ &= x^{r+s}k(\frac{1}{x})g(\frac{1}{x}), \text{ donde } m = r+s, \text{ para algún } s > 0, \\ &= x^s k(\frac{1}{x})(x^r g(\frac{1}{x})) = x^s k(\frac{1}{x})g(x). \end{aligned}$$

Por lo tanto, $p^*(x) \in C$, es decir, es reversible. ∎

Es fácil verificar que si $f(x), g(x)$ son dos polinomios en $GF(4)[x]$, con $grad(f(x)) = m$ y $grad(g(x)) = k$, con $m \geq k$, entonces,

1. $[f(x)g(x)]^* = f^*(x)g^*(x)$.
2. $[f(x) + g(x)]^* = f^*(x) + x^{m-k}g^*(x)$.

En seguida se enunciarán algunos resultados que relacionan las características de los polinomios generadores de un código con la forma de dicho código y las propiedades que cumple.

Teorema 5.14 *Sea $C = \langle \omega p(x) + q(x)\rangle$ un código cíclico aditivo de longitud n sobre $GF(4)$ con $grad(p(x)) = r$. Entonces C es reversible si y sólo si $p(x)$ es un polinomio binario auto-recíproco y $C = \langle \omega p(x)\rangle$ ó $C = \langle \bar{\omega} p(x)\rangle$.*

Teorema 5.15 *Sea $C = \langle \omega p(x), r(x)\rangle$ ó $C = \langle \bar{\omega} p(x), r(x)\rangle$, un código cíclico aditivo de longitud n sobre $GF(4)$. El código C es complemento reversible si y sólo si $p(x)$, $r(x)$ son auto-recíprocos y $r(x)$ no es múltiplo de $x - 1$.*

Teorema 5.16 *Sea $C = \langle \omega p(x) + q(x), r(x) \rangle$ un código cíclico aditivo de longitud n sobre $GF(4)$, con $q(x) \neq 0$. El código C es un código cíclico aditivo complemento reversible si y sólo si $p(x)$ es auto-recíproco, $r(x)$ es auto-recíproco y no es múltiplo de $x - 1$, y*

1. *Si $grad(q(x)) = grad(p(x))$, entonces $q(x)$ es auto-recíproco.*
2. *Si $grad(q(x)) < grad(p(x))$, entonces $r(x)$ divide a*
$$[x^{grad(p(x))-grad(q(x))}q^*(x) + q(x)].$$
3. *Si $grad(q(x)) > grad(p(x))$, entonces $r(x)$ divide a*
$$[x^{grad(q(x))-grad(p(x))}q(x) + q^*(x)].$$

Puede consultar las pruebas de estos resultados en [5].

6. Códigos de longitud 7

En esta última sección usaremos la teoría desarrollada para estudiar códigos aditivos complemento cíclicos reversibles sobre $GF(4)$, es decir, estudiaremos códigos que tienen las características necesarias para codificar cadenas de ADN.

Los códigos complemento cíclicos reversibles cumplen las siguientes restricciones (Vea [1]):

- *Restricción de Hamming:* Para dos diferentes palabras código u y $u' \in C$, $H(u, u') \geq d$, donde d es la distancia del código C. Esta restricción está destinada a limitar la hibridación no específica entre una palabra código y un complemento Watson-Crick de una palabra diferente.
- *Restricción del complemento reverso:* Para algún par de palabras código, u y $u' \in C$, (las cuales pueden ser iguales), $H(u^c, (u')^r) \geq d$. Esta restricción pretende limitar la hibridación no específica entre una palabra código y el reverso de otra palabra código.

Presentamos algunos ejemplos de códigos de longitud 7.

Consideremos el polinomio $x^7 - 1$, sus divisores son:

$$\begin{aligned}
p_0(x) &= 1,\\
p_1(x) &= x + 1,\\
p_2(x) &= x^3 + x^2 + 1,\\
p_3(x) &= x^3 + x + 1,\\
p_4(x) &= x^4 + x^3 + x^2 + 1 = (x + 1)(x^3 + x + 1),\\
p_5(x) &= x^4 + x^2 + x + 1 = (x + 1)(x^3 + x^2 + 1),\\
p_6(x) &= x^6 + x^5 + x^4 + x^3 + x^2 + x + 1 = (x^3 + x^2 + 1)(x^3 + x + 1).
\end{aligned}$$

Calculemos el recíproco de estos polinomios:

$$\begin{aligned}
p_0^*(x) &= 1,\\
p_1^*(x) &= x+1,\\
p_2^*(x) &= x^3+x+1,\\
p_3^*(x) &= x^3+x^2+1,\\
p_4^*(x) &= x^4+x^2+x+1,\\
p_5^*(x) &= x^4+x^3+x^2+1,\\
p_6^*(x) &= x^6+x^5+x^4+x^3+x^2+x+1.
\end{aligned}$$

De ellos notamos que los polinomios p_0, p_1, p_6 son auto-recíprocos. Utilizando estos polinomios para generar códigos aditivos que cumplan las condiciones del Teorema 5.16, encontramos que los códigos:

$$\begin{aligned}
C_1 &= \langle \omega+1, x^6+x^5+x^4+x^3+x^2+x+1\rangle,\\
C_2 &= \langle \omega+(x^4+x^3), x^6+x^5+x^4+x^3+x^2+x+1\rangle,\\
C_3 &= \langle \omega+(x^4+x^3+1), x^6+x^5+x^4+x^3+x^2+x+1\rangle,\\
C_4 &= \langle \omega+(x^4+x^2), x^6+x^5+x^4+x^3+x^2+x+1\rangle,\\
C_5 &= \langle \omega+(x^5+x^4+x^3+x^2), x^6+x^5+x^4+x^3+x^2+x+1\rangle,\\
C_6 &= \langle \omega+(x^5+x^2+1), x^6+x^5+x^4+x^3+x^2+x+1\rangle,\\
C_7 &= \langle \omega+(x^5+x^4+x^3+x^2+1), x^6+x^5+x^4+x^3+x^2+x+1\rangle.
\end{aligned}$$

Son códigos aditivos complemento cíclicos reversibles, además, analizando sus elementos generadores, se encontró que tienen distancias:

$$\begin{aligned}
d(C_1) &= 1,\\
d(C_2) &= 3,\\
d(C_3) &= 3,\\
d(C_4) &= 3,\\
d(C_5) &= 3,\\
d(C_6) &= 3,\\
d(C_7) &= 3.
\end{aligned}$$

Sin embargo, nos interesan los códigos que tienen mayor distancia, pues ellos permiten identificar e incluso corregir errores, así que no se considerará al código C_1.

Se llegó a la conclusión de que los códigos

$$\begin{aligned}
C_2 &= \langle \omega+(x^4+x^3), x^6+x^5+x^4+x^3+x^2+x+1\rangle,\\
C_3 &= \langle \omega+(x^4+x^3+1), x^6+x^5+x^4+x^3+x^2+x+1\rangle,\\
C_4 &= \langle \omega+(x^4+x^2), x^6+x^5+x^4+x^3+x^2+x+1\rangle,\\
C_5 &= \langle \omega+(x^5+x^4+x^3+x^2), x^6+x^5+x^4+x^3+x^2+x+1\rangle,\\
C_6 &= \langle \omega+(x^5+x^2+1), x^6+x^5+x^4+x^3+x^2+x+1\rangle,\\
C_7 &= \langle \omega+(x^5+x^4+x^3+x^2+1), x^6+x^5+x^4+x^3+x^2+x+1\rangle,
\end{aligned}$$

por el Teorema 5.12, tienen dimensión $2n - grad(p(x)) - grad(r(x)) = 2(7) - 0 - 6 = 8$, y por lo tanto, tienen tamaño $M = 2^8 = 256$. Con esto, se puede asegurar que estos códigos, mejoran las codificaciones para los algoritmos de ADN, pues tienen distancia 3 y por lo tanto son capaces de detectar hasta 2 errores, y corregir automáticamente 1 error, de acuerdo con el Teorema 3.6. Las codificaciones que se han usado con esta distancia y longitud han sido de a lo más 180 palabras (Vea [1],[2],[7], [9]), al tener la misma distancia, es mejor utilizar códigos que tienen 256 palabras, pues evita tener hibridaciones no deseables entre las hebras de ADN.

Para poder realizar los algoritmos para las aplicaciones de la computación del ADN que se mencionaron en la sección 2 es necesario contar con una codificación del ADN que sea útil y que dé como resultado el mínimo de enlaces indeseables posibles, esto se logra con las condiciones que se presentan en este trabajo, por lo que se puede concluir que los códigos de longitud 7 que presentamos mejoran las codificaciones que se habían utilizado con anterioridad.

Conclusión

En este artículo se presentaron condiciones para los códigos cíclicos sobre $GF(4)$ que permiten dar una mejor codificación para los algoritmos de ADN, basándonos en el artículo de Abualrub [1], se describieron los elementos básicos que se deben considerar para crear códigos que sean útiles para representar cadenas de ADN y finalmente se presentaron ejemplos de códigos aditivos complemento cíclicos reversibles de longitud 7 y tamaño 256 que son útiles para la computación del ADN, puesto que cumplen con la *Restricción de Hamming* y la *Restricción del complemento reverso.*

Agradecimientos

A los revisores de este escrito, quienes con sus sugerencias contribuyeron a obtener una versión mejorada de nuestro trabajo.

Referencias

[1] T. Abualrub, A. Ghrayeb, and X. N. Zeng, "Construction of cyclic codes over gf (4) for dna computing," *Journal of the Franklin Institute*, vol. 343, no. 4-5, pp. 448–457, 2006. DOI: 10.1016/j.jfranklin.2006.02.009

[2] R. Deaton, R. C. Murphy, J. Rose, M. Garzon, D. R. Franceschetti, and S. Stevens, "A dna based implementation of an evolutionary search for good encodings for dna computation," in *Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC'97)*. IEEE, 1997. DOI: 10.1109/ICEC.1997.592311 pp. 267–271.

[3] H. Eghdami and M. Darehmiraki, "Application of dna computing in graph theory," *Artificial Intelligence Review*, vol. 38, pp. 223–235, 2012. DOI: 10.1007/s10462-011-9247-5

[4] X. Liu, Y. Li, and J. Xu, "Solving minimum spanning tree problem with dna computing," *Journal of Electronics (China)*, vol. 22, pp. 112–117, 2005. DOI: 10.1007/BF02688136

[5] N. A. G. Martınez, "Códigos cıclicos basados en campos finitos para desarrollar algoritmos de adn," Ph.D. dissertation, UNIVERSIDAD TECNOLOGICA DE LA MIXTECA, 2019. [Online]. Available: http://jupiter.utm.mx/~tesis_dig/13800.pdf

[6] A. Neubauer, J. Freudenberger, and V. Kuhn, *Coding theory: algorithms, architectures and applications*. John Wiley & Sons, 2007.

[7] V. V. Rykov, A. J. Macula, D. C. Torney, and P. White, "Dna sequences and quaternary cyclic codes," in *Proceedings. 2001 IEEE International Symposium on Information Theory (IEEE Cat. No. 01CH37252)*. IEEE, 2001. DOI: 10.1109/ISIT.2001.936111 p. 248.

[8] Syngenta México. Qué es ADN? [Online]. Available: https://www.syngenta.com.mx/que-es-adn

[9] D. C. Tulpan, H. H. Hoos, and A. E. Condon, "Stochastic local search algorithms for dna word design," in *DNA Computing: 8th International Workshop on DNA-Based Computers, DNA8 Sapporo, Japan, June 10–13, 2002 Revised Papers 8*. Springer, 2003. DOI: 10.1007/3-540-36440-4_20 pp. 229–241.

[10] Universidad de Granada. Las matemáticas en las comunicaciones. [Online]. Available: https://www.ugr.es/~anillos/textos/pdf/2010/EXPO-2.Matematicas%20comunicaciones/101.htm

Como citar este artículo: N. A. González Martínez y A. Maceda Méndez, "Códigos cíclicos para desarrollar algoritmos de ADN", Sahuarus. Revista Electrónica de Matemática, vol. 7, no. 2, pp. 14–28, 2023. https://doi.org/10.36788/sah.v7i2.141

Los contenidos de este artículo están bajo una licencia de Creative Commons Atribución No Comercial - Sin Obra Derivada 4.0 Internacional