BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS DE LA COMPUTACIÓN

Localización de Motivos de secuencias de ADN usando un algoritmo genético

TESIS

Para obtener el título de:

Licenciatura en Ciencias de la Computación

Presenta:

Yesenia Guadalupe Romero Sánchez

Asesor:

M. C. Marcela Rivera Martínez

Febrero 2022

A mis padres, lo más importante en mi vida.

AGRACEDIMIENTOS

Quiero agradecer a mi asesora de tesis la maestra Marcela Rivera, gracias por el tiempo y el apoyo dedicado a este trabajo, y por tenerme paciencia, muchas gracias.

A mis padres, gracias por el apoyo y el esfuerzo que me han brindado durante toda mi vida, ya que sin ustedes no lograría nada de lo que soy hoy en día, gracias por todo lo que me han dado.

A mi hermana por su apoyo y por ser la única que logra entender mis locuras.

Gracias al PRODEP 511-6-2020-7851 por brindarme el apoyo para el desarrollo de esta tesis.

ÍNDICE

1. INTRODUCCIÓN

La presente tesis tiene como objetivo implementar un algoritmo genético para la búsqueda de motivos en secuencias de ADN, con la finalidad de mejorar la búsqueda, haciéndola más eficiente y precisa, así como facilitar su uso en diferentes investigaciones.

La evolución genética siempre ha sido un tema de relevancia desde principios del siglo XX, cuando se comenzó a hablar de partículas, factores, caracteres y genes, se comenzaron a esclarecer las bases moleculares, es decir, cómo estaban compuestos los lugares que contenían esa información y dónde estaba localizada.

A pesar que se trata de una disciplina científica con sólo unos 150 años de desarrollo, el interés en los fenómenos biológicos relacionados con la Genética puede remontarse a la Era Neolítica (alrededor de 10.700 a 9.400 AC) [1].

El proceso de traspaso de información o de material genético de una generación a otra, depende completamente de cómo la célula crece y se divide [2].

Gregor Johann Mendel establece en 1865 las leyes de la herencia, fundando la disciplina científica de la genética, es por esto que se le conoce como el padre de la genética [1].

Posteriormente vino el descubrimiento de la estructura del Ácido Desoxirribonucleico (ADN), Watson y Crick postularon un modelo preciso para la estructura tridimensional del ADN, en el que explicaban cómo la información genética podía replicarse con exactitud. El modelo proponía que dos cadenas o hebras de polinucleótidos se hallaban enrolladas en forma de hélice alrededor de un mismo eje, constituyendo así una doble hélice. El enrollamiento de ambas cadenas es tal que no pueden separarse sin desenrollarse.

Después del establecimiento del modelo de doble hélice (ver figura 1.1), los ácidos nucleicos se consideran los constituyentes químicos de los genes y los que regulan la reproducción celular y la biosíntesis de proteínas. Los ácidos nucleicos son los portadores de la información genética celular. Hay dos tipos de ácidos nucleicos: el ácido desoxirribonucleico (ADN) y el ácido ribonucleico (ARN).

Las bases se encuentran en el interior de la doble hélice, con sus planos paralelos entre sí, y perpendiculares al eje de la doble hélice. Las bases de una de las hebras están apareadas en los mismos planos con las bases de la otra hebra. El apareamiento de las bases es de tal forma que sólo encajan en la estructura determinados pares de bases que pueden ligarse entre sí, por enlaces de hidrógeno. Los pares permitidos son: adenina-timina y guanina-citosina, precisamente los pares de bases que muestran equivalencia en el ADN.

El ADN es el constituyente de los genes y por ello el portador de la información genética. Con la replicación del ADN en el proceso celular los caracteres genéticos pasan a las células descendientes que reproducen cadenas de ADN con una estructura

complementaria de las originales, lo que permite la obtención de dobles cadenas de ADN en las células hijas, idénticas a las parentales de la célula original.

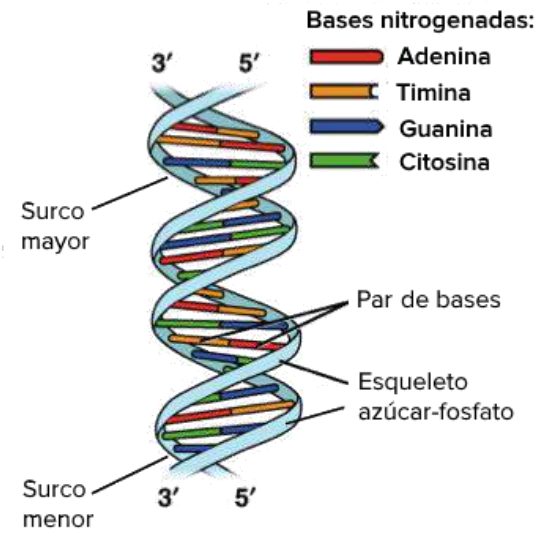


Figura 1.1: Modelo de Doble Hélice del ADN.

El ARN tiene como misión fundamental ser intermediario en la biosíntesis de proteínas en dos procesos consecutivos. El primero es la transcripción del código genético (secuencia de bases nitrogenadas) del ADN celular a la secuencia de bases complementaria en el ARN formado a partir de la unión de nucleótidos sobre el molde de ADN. El segundo proceso, es la traducción del código genético (secuencia de tres bases) del ADN, o su complementario en el ARN formado, a la secuencia de aminoácidos respectivos, en la biosíntesis de proteínas.

Las bases nitrogenadas se unen al carbono-1 (anomérico) del azúcar en ambos ácidos nucleicos. El ADN lleva Adenina (A), Guanina (G), Timina (T) y Citosina (C) (ver figura 1.2) [3].

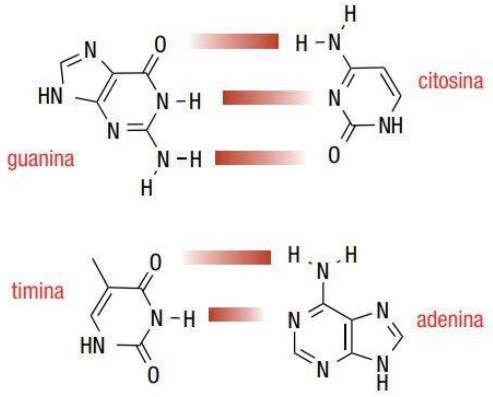


Figura 1.2: Bases nitrogenadas que componen el ADN.

El descubrimiento de motivos es uno de los problemas del análisis de secuencias de ADN, los motivos representan secuencias conservadas que pueden ser biológicamente significativas, el descubrimiento de motivos consiste en encontrar sitios de unión en aminoácidos, encontrando reguladores de información dentro de secuencias de ADN o ARN [5].

Un motivo de ADN se define como un patrón de secuencia de ácido nucleico que tiene algún significado biológico como lo son sitios de unión de ADN para una proteína reguladora, es decir, un factor de transcripción. Normalmente dicho patrón es bastante corto (5 a 20 pares de bases (pb) de largo) y se sabe que es recurrente en diferentes genes o varias veces dentro de un gen [6].

Los motivos representan zonas conservadas entre las secuencias que suelen asociarse a características funcionales del grupo de secuencias. Una vez se ha construido un motivo o patrón de un grupo de secuencias puede utilizarse, para asociar una nueva secuencia con la familia de secuencias que lo ha generado (si el motivo que se presenta es de la familia, puede que comparta sus funciones), para buscar secuencias que pertenezcan a aquella familia [7].

La secuenciación del ADN a gran escala de varios organismos ha resultado en la generación de una gran cantidad de datos biológicos y por lo tanto, siempre hay una creciente necesidad de desarrollar técnicas computacionales que pueden ayudar a encontrar información útil entre todos los datos. Descubrir motivos implica determinar secuencias cortas y significativas que pueden repetirse sobre muchas secuencias en varias especies.

La identificación de motivos se está volviendo muy importante, porque representan secuencias conservadas que pueden ser biológicamente significativas. Algunas de las áreas donde el descubrimiento de motivos puede ser útil incluyen encontrar sitios de unión en aminoácidos, encontrar reguladores de información dentro de secuencias de

ADN o ARN, buscar información de empalme de proteínas y dominios. Los motivos pueden representar patrones que activan o inhiben el proceso de transcripción y son responsables de regular la expresión génica [5].

Reconocer motivos no es obtenerlos, sino identificarlos en otras secuencias, generalmente con el objetivo de establecer inferencias en cuanto a estructura y función.

Existen ciertos desafíos en la identificación de motivos, de los cuales incluyen [5]:

- Los motivos nunca son exactamente iguales a las actuales secuencias conservadas, siempre hay mucha variabilidad de secuencia presente con respecto a un solo motivo.
- Los motivos son señales muy cortas en comparación con el tamaño de la secuencia de ADN en consideración.
- Las secuencias reguladoras que contienen los motivos a veces se pueden ubicar muy lejos de las regiones de codificación que regulan, esto hace que sea difícil determinar la porción de la secuencia de ADN que debería ser analizada.
- Las secuencias reguladoras pueden, a veces, estar presentes en el hilo opuesto de la secuencia de codificación que regulan.
- El problema del descubrimiento de motivos es un problema NP, esto quiere decir, que no hay solución de tiempo polinómico para el descubrimiento de motivos.

Un factor importante dentro del problema de encontrar motivos es identificar elementos reguladores, especialmente los sitios de unión en el ADN para la transcripción de factores.

La técnica de descubrimiento de motivos consta de tres etapas principales [8]:

A. Pre-procesamiento
   Se preparan las secuencias de ADN para un movimiento preciso, se seleccionan muchas secuencias posibles como objetivos que contengan motivos, se mantienen las secuencias lo más cortas posibles y se eliminan las improbables que contengan motivos. El paso de ensamblado se realiza agrupando las secuencias de entrada basado en alguna información y se extraen las secuencias apareadas en una base de datos en secuencia apropiada. Después, limpia las secuencias de entrada para enmascarar o eliminar secuencias de confusión necesarias.

B. Descubrimiento
   La etapa intermedia es el enfoque de descubrimiento de motivos, eso comienza representando las secuencias. Existen dos formas de representar los motivos: cadenas de consenso y matrices de peso específicas de posición (PWM). Las cadenas de consenso tienen la misma longitud de un motivo de secuencia de ADN, eso permite degenerar símbolos en una cadena usando la nomenclatura IUPAC (Unión Internacional de Química Pura y Aplicada); mientras PWM es una matriz de $4xm$ donde m es la longitud del motivo, cada posición de la matriz representa la probabilidad de cada nucleótido en cada índice de posición del

motivo. Después de la representación del motivo, la función objetivo adecuada es determinada y finalmente el algoritmo de búsqueda apropiado es aplicado.

C. Post-procesamiento
   El post-procesamiento evalúa los motivos resultantes. En esta parte se aplican los diferentes métodos de extracción de motivos de secuencias.

Esta investigación consta de seis capítulos, a saber. En el capítulo uno se hace una introducción acerca del objetivo de la tesis y se presentan los antecedentes del código genético así como el descubrimiento de los motivos en el ADN. En el capítulo dos se explican los diferentes métodos que existen para la búsqueda de motivos en secuencias de ADN. En el capítulo tres se describe la conformación de los algoritmos genéticos. En el capítulo cuatro se desarrolla la explicación del algoritmo implementado. En el capítulo cinco se muestran las distintas pruebas realizadas al algoritmo. En el capítulo seis se plantean las conclusiones del trabajo realizado, y finalmente se listan las referencias.

2. ESTADO DEL ARTE

En este capítulo se mostrarán los métodos que existen para la búsqueda de motivos en secuencias de ADN.

Existen diferentes métodos para resolver esta problemática, en general, para resolver este problema existen dos principales grupos de métodos: los de enfoque de enumeración y de optimización probabilística. Los métodos de enfoque de enumeración buscan por consensos de secuencias, los motivos se predicen en función de la enumeración de palabras y similitudes de palabras computacionales, también se le conoce como enfoque de enumeración de palabras, el segundo grupo son los métodos de optimización probabilística [4].

2.1 Método por enfoque de enumeración

El método por enfoque de enumeración busca el consenso de secuencias, los motivos se predicen en función de la enumeración de palabras y cálculo de similitud de palabras, este enfoque a veces se le llama enfoque de enumeración de palabras para resolver el problema de motivo (*l*, *d*) con longitud (*l*) y un número máximo de desajustes (*d*).

Los algoritmos basados en enfoque de enumeración de palabras buscan exhaustivamente el conjunto del espacio de búsqueda para determinar cuáles aparecen con sustituciones posibles y, por lo tanto, normalmente localizan un óptimo global. Sin embargo, esto también significa que son algoritmos de tiempo exponencial que requieren un mayor tiempo para la detección y para el manejo de docenas de secuencias, por lo que solo son adecuadas para motivos cortos. Además, estos algoritmos requieren muchos parámetros determinados por los usuarios, como la longitud del motivo, el número de discrepancias permitidas y un mínimo número de secuencias en las que tiene que aparecer el motivo [8].

2.1.1 Enumeración simple de palabras

Este método se basa en una simple enumeración de palabras, como se mencionó anteriormente, uno de los algoritmos que existe enfocados en este método es DREME (Discriminative Regular Expression Motif Elicitation).

DREME encuentra rápidamente motivos relativamente cortos (hasta 8 posiciones). La entrada a DREME es uno o dos conjuntos de secuencias y un umbral de significación. Las secuencias de control deben tener aproximadamente la misma longitud que las secuencias primarias. El algoritmo crea un ciclo externo, realiza una búsqueda heurística de motivos, determina el motivo significativo, lo informa y borra todas sus ocurrencias en

los conjuntos de datos de entrada. El bucle exterior se repite hasta que ningún motivo nuevo tenga un valor menor que el dado en el umbral de significación.

Alcanza su alta velocidad al restringir su búsqueda a expresiones regulares basadas en los símbolos disponibles en el alfabeto y al usar una estimación heurística de la significancia estadística de motivos generalizados [9].

2.1.2 Método basado en agrupaciones

Para el método basado en agrupaciones se tiene al algoritmo CisFinder, este genera una lista completa de motivos enriquecidos en un conjunto de secuencias de ADN y las describe con matrices de frecuencia de posición (PFM). CisFinder es uno de los algoritmos que mejor ejemplifican este método.

El algoritmo CisFinder se basa en la detección de representación de palabras cortas (secuencias de nucleótidos) en una secuencia y las agrupa. A diferencia del oligo-análisis (RSAT) [10], que también está basado en el mismo concepto pero agrupa exactamente palabras, CisFinder agrupa las PFM (matrices de frecuencia de posición) que representan motivos de encuadernación con más precisión exacta de palabras. Analiza palabras con espacios en blanco y expande PFM sobre los espacios y flanqueando regiones. Utiliza secuencias de control reales para comparar contra secuencias de prueba, esto ayuda a procesar regiones repetidas, porque los motivos que son específicos en repetir las secuencias se esperan que sean igualmente abundantes en los conjuntos de prueba y control de secuencias.

CisFinder está diseñado para llevar a cabo exhaustivas búsquedas de todos los motivos de ADN sobrerrepresentados en una sola carrera [11].

2.1.3 Método basado en árboles

La tercera clase de los métodos es una búsqueda basada en árboles para acelerar la enumeración de palabras, un ejemplo es el algoritmo Weeder basado en patrones de coincidencia de conteo con desajustes específicos y más extremos.

En este algoritmo, los motivos se representan mediante secuencia de consenso, y basado en la diferencia entre los *k-mers* de las secuencias de entrada y el consenso bajo de un limitado número de sustituciones *k-mers*, se ensamblan y cada grupo se evalúa con una medida específica de significación. El árbol de sufijos construido se propuso en el algoritmo FMotif [12] para encontrar motivos largos (*l*, *d*) en grandes secuencias de ADN bajo ZOMOPS (zero, one or multiple ocurrences of the motifs instances per sequence) por su significado en ingles cero, uno o múltiples ocurrencias de las instancias de un motivo por secuencia. Este árbol propuesto es más rápido que el árbol de sufijos estándar, ya

que evita una gran cantidad de escaneos repetidos de secuencias en el árbol de sufijos [13].

2.1.4 Método basado en teoría de grafos

El método basado en teoría de grafos representa un motivo en instancia, como un clic; el grafo *G* se construye representando cada *l-mer* en las secuencias de entrada por vértice y el borde entre un par de vértices, que representan un par de *l-mer* en diferentes secuencias de entrada, que tienen la distancia de Hamming entre las subcadenas que sean menor o igual a 2d. Luego, se buscan clics de tamaño *N* en este grafo [8].

Uno de los métodos más populares de teoría de grafos es Winnower, este algoritmo detecta motivos borrosos en secuencias de ADN ricas en señales de unión a proteínas. El algoritmo encuentra tales motivos si varias copias mutadas del motivo (es decir, en señales) están presentes en la secuencia de ADN en suficiente abundancia. Este mejora sustancialmente la sensibilidad del método winnower de Pevzner y Szeby poniendo una restricción de consenso, lo que le permite detectar señales mucho más débiles [14].

2.1.5 Método basado en Hash

Karl Buhler desarrolló algoritmos de proyección aleatoria para PMP que proyecta cada *l-mer* en la entrada de datos de un espacio más pequeño mediante hash. Inicialmente, se tiene una proyección del espacio *l*-dimensional en una sub-dimensión *k*. Se desarrolla el espacio para todas las subsecuencias en el conjunto de entrada, y la proyección aleatoria se construye eligiendo rangos de dominio *k* posiciones, desde la posición *l*. Usando esta proyección, cada *l-mer* tiene un hash en su cubo correspondiente. Después de las proyecciones, cada cubo contiene *l-mer* en más de un umbral y esto se llama depósito calificado. El hash aleatorio se repite *n* veces para asegurar que el cubo aparezca al menos una vez. Finalmente, el perfil para cada uno de ellos debe calcularse para obtener el *l-mer* en la secuencia que se representó como consecuencia [15].

2.1.6 Candidatos fijos y métodos basados en candidatos modificados

La sexta clase son candidatos fijos que seleccionan motivos candidatos desde las secuencias de entrada y los utiliza para la lectura de motivos, mientras se modifica la séptima clase de motivos que selecciona un candidato de la secuencia de entrada y lo modifica letra a letra.

Hay un algoritmo propuesto llamado Ref-Seleccione, para seleccionar secuencias de referencia para PMP. Las secuencias de referencia son las secuencias que no contienen

instancias de motivo, por lo que este método intenta seleccionar las secuencias de referencia que generan una pequeña cantidad de motivos candidatos como sea posible.

El algoritmo consta de dos pasos; en primer lugar, para cada dos secuencias en los datos de entrada *D*, el número de motivos candidatos generados deben calcularse utilizando la distancia de Hamming entre cada dos *l-mers*. Luego, el conjunto con los motivos candidatos, los más pequeños posible, se seleccionan como referencia de dicho conjunto [16].

2.2 Enfoque probabilístico

El enfoque probabilístico ocupa un lugar importante en los últimos desarrollos en inteligencia artificial, robótica y aprendizaje automático. Las leyes de la probabilidad permiten una representación lógica de la racionalidad, que también sigue el comportamiento humano y que se basa en la probabilidad de aceptar un postulado como cierto a partir de recompensas futuras [17].

2.2.1 Enfoque determinista

Uno de los mejores ejemplos de aplicación al enfoque determinista es el algoritmo MEME (Multiple EM for Motif Elicitation), su objetivo principal es encontrar un motivo inicial y después aplicar los pasos de expectativa y maximización; el primero estima los valores de algún conjunto de incógnitas basadas en un conjunto de parámetros, el segundo paso utiliza esos valores estimados para refinar los parámetros en varias iteraciones [8].

MEME usa una función objetivo en motivos para seleccionar el mejor motivo, la función objetivo se basa en la significación estadística de la razón logarítmica de probabilidad de las apariciones del motivo. El valor *E* del motivo es una estimación del número de motivos (con el mismo ancho y número de ocurrencias) que tendrían una razón logarítmica de probabilidad igual o mayor si las secuencias de entrada se hubieran generado aleatoriamente de acuerdo con la porción de orden 0 del modelo de fondo [18].

2.2.2 Enfoque estocástico

Dentro del enfoque estocástico se encuentra el Muestreo de Gibbs, el cual es un algoritmo para generar una muestra aleatoria a partir de la distribución de probabilidad conjunta de dos o más variables aleatorias. El proceso es una búsqueda aleatoria a lo largo del espacio de parámetros, la búsqueda inicia en un punto arbitrario y en cada momento el siguiente paso depende únicamente de la posición actual [19].

El algoritmo de muestreo de Gibbs para la detección de motivos sigue los siguientes pasos [8]:

1. Inicialización aleatoria de posiciones de motivo en las *N* secuencias de entrada con la suposición de la presencia de un motivo por secuencia.
2. Elegir una secuencia al azar.
3. Calcular PWM para las otras *N-1* secuencias, usando posiciones iniciales de motivos y probabilidades de fondo para cada base empleando las posiciones sin motivo.
4. Calcular la probabilidad de cada posible ubicación de motivo en la secuencia eliminada utilizando PWM y probabilidades de fondo.
5. Para la secuencia eliminada, elija una nueva posición de inicio según el paso 4.
6. Los pasos 2-5 deben repetirse hasta que los valores en el PWM no mejoren o se haya alcanzado el número máximo de iteraciones.

2.2.3 Enfoque avanzado

Este método se enfoca en algoritmos basados en el enfoque bayesiano con la cadena de Markov Monte Carlo, uno de los desarrollados es LOGOS (Integrated Local and Global motif sequence model) que proporciona un marco de principios para desarrollar, modularizar, extender y computar modelos de motivos expresivos para el análisis de secuencias de biopolímeros complejos. LOGOS consta de dos submodelos que interactúan: HMDM, un modelo de alineación local que captura el conocimiento biológico previo y la dependencia posicional dentro de la estructura local del motivo; y HMM, un modelo de distribución de motivos global que modela frecuencias y dependencias de ocurrencias de motivos, los parámetros del modelo se pueden ajustar utilizando motivos de entrenamiento dentro de un marco empírico bayesiano [20].

3. ALGORITMOS GENÉTICOS

En este capítulo se abordará el origen y explicación de los algoritmos genéticos.

Los Algoritmos Genéticos hacen evolucionar una población de individuos, o conjunto de soluciones posibles del problema, sometiéndola a acciones aleatorias semejantes a las que actúan en la evolución biológica tales como mutaciones y recombinaciones genéticas; así como también a una selección de acuerdo con algún criterio, en función del cual se decide cuáles son los individuos más adaptados, que sobreviven, y cuáles los menos aptos, que son descartados [21].

3.1 Origen y Antecedentes

El desarrollo de los Algoritmos Genéticos se debe a John Holland (ver figura 3.1), investigador de la Universidad de Michigan. A finales de la década de los 60 desarrolló una técnica que imitaba en su funcionamiento a la selección natural. Aunque originalmente esta técnica recibió el nombre de “planes reproductivos”, a raíz de la publicación en 1975 de su libro “Adaptation in Natural and Artificial Systems” se conoce principalmente con el nombre de Algoritmos Genéticos.

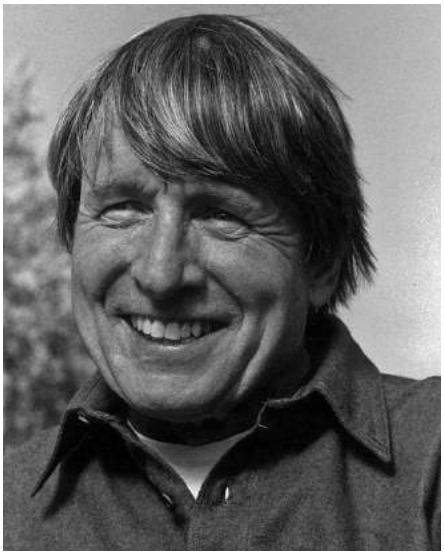


Figura 3.1: John Henry Holland.

A grandes rasgos, un Algoritmo Genético consiste de una población de soluciones codificadas de forma similar a cromosomas. Cada uno de estos cromosomas tendrá asociado un ajuste, valor de bondad o fitness, que cuantifica su validez como solución al problema. En función de este valor se le darán más o menos oportunidades de

reproducción. Además, con cierta probabilidad se realizarán mutaciones de estos cromosomas, este proceso hará posible que los individuos genéticos tiendan hacia las soluciones a un problema dado, aunque las condiciones del espacio de búsqueda varíen con el transcurso del tiempo.

Las bases de las Estrategias de Evolución fueron apuntadas en 1973 por Rechenberg en su obra “Evolutions strategie: Optimierung Technisher Systeme nach Prinzipien der Biologischen Evolution” (Rechenberg, 1973). Las dos Estrategias de Evolución más empleadas son la (μ+λ)-ES y la (μ, λ)-ES. En la primera de ellas un total de μ padres producen λ descendientes, reduciéndose nuevamente la población a μ individuos (los padres de la siguiente generación) por selección de los mejores individuos. De esta manera los padres sobreviven hasta que son reemplazados por hijos mejores que ellos. En la (μ, λ)-ES la descendencia reemplaza directamente a los padres, sin hacer ningún tipo de comprobación.

Cualquier solución potencial a un problema, puede ser presentada dando valores a una serie de parámetros. El conjunto de todos los parámetros (genes en la terminología de Algoritmos Genéticos) se codifica en una cadena de valores denominada cromosoma (ver figura 3.2). El conjunto de los parámetros representado por un cromosoma particular recibe el nombre de genotipo, ese contiene la información necesaria para la construcción del organismo, es decir, la solución real al problema, denominada fenotipo.

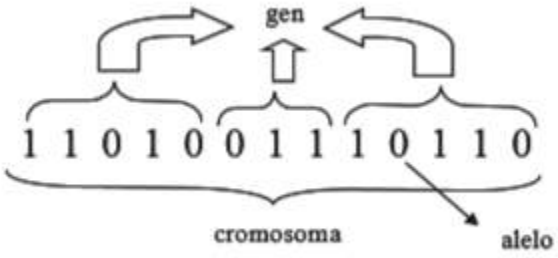


Figura 3.2: Individuo genético binario.

Desde los primeros trabajos de John Holland, la codificación suele hacerse mediante valores binarios. Se asigna un determinado número de bits a cada parámetro y se realiza una discretización de la variable representada por cada gen. El número de bits asignados dependerá del grado de ajuste que se desee alcanzar. Evidentemente, no todos los parámetros tienen por qué estar codificados con el mismo número de bits. Cada uno de los bits pertenecientes a un gen suele recibir el nombre de alelo.

Sin embargo, también existen representaciones que codifican directamente cada parámetro con un valor entero, real o en punto flotante. A pesar de que se acusa a estas representaciones de degradar el paralelismo implícito de las representaciones binarias,

permiten el desarrollo de operadores genéticos más específicos al campo de aplicación del Algoritmo Genético [22].

3.2 Estructura de los algoritmos genéticos

Los Algoritmos Genéticos trabajan sobre una población de individuos, cada uno de ellos representa una posible solución al problema que se desea resolver. Todo individuo tiene asociado un ajuste de acuerdo a la bondad con respecto al problema de la solución que representa. Una generación se obtiene a partir de la anterior por medio de los operadores de reproducción, existen 2 tipos:

- Cruce: Se trata de una reproducción de tipo sexual. Se genera una descendencia a partir del mismo número de individuos (generalmente 2) de la generación anterior.
- Copia: Se trata de una reproducción de tipo asexual. Un determinado número de individuos pasa sin sufrir ninguna variación directamente a la siguiente generación.

Algoritmo 3.1: Funcionamiento de un Algoritmo Genético.

```
1  Inicializar población actual
2  MIENTRAS no se cumpla el criterio de evaluación
3      crear población temporal vacía
4      SI elitismo: copiar en población temporal mejores individuos
5      MIENTRAS población temporal no llena
6          seleccionar padres
7          cruzar padres con probabilidad Pc
8          SI se ha producido el cruce
9              mutar uno de los descendientes (prob. Pm)
10             evaluar descendientes
11             añadir descendientes a la población temporal
12         SINO
13             añadir padres a la población
14         FIN SI
15     FIN MIENTRAS
16     aumentar contador generaciones
17     establecer como nueva población actual la población temporal
18 FIN MIENTRAS
```

El funcionamiento genérico de un Algoritmo Genético se muestra en el pseudocódigo (ver Algoritmo 3.1), Si desea optarse por una estrategia elitista, los mejores individuos de cada generación se copian siempre en la población temporal, para evitar su pérdida. A

continuación, comienza a generarse la nueva población en base a la aplicación de los operadores genéticos de cruce y/o copia. Una vez generados los nuevos individuos se realiza la mutación con una probabilidad *Pm*. La probabilidad de mutación suele ser muy baja, por lo general entre el 0.5% y el 2%. Se sale de este proceso cuando se alcanza alguno de los criterios de parada fijados, los más usuales suelen ser:

- Los mejores individuos de la población representan soluciones suficientemente buenas para el problema que se desea resolver.
- La población ha convergido. Un gen ha convergido cuando el 95% de la población tiene el mismo valor para él, en el caso de trabajar con codificaciones binarias, o valores dentro de un rango especificado en el caso de trabajar con otro tipo de codificaciones. Una vez que todos los genes alcanzan la convergencia se dice que la población ha convergido. Cuando esto ocurre la media de bondad de la población se aproxima a la bondad del mejor individuo.
- Se ha alcanzado el número de generaciones máximo especificado.

3.2.1 Operadores genéticos

Para el paso de una generación a la siguiente se aplican una serie de operadores genéticos. Los más empleados son los operadores de selección, cruce, copia y mutación. En el caso de no trabajar con una población intermedia temporal también cobran relevancia los algoritmos de reemplazo. A continuación, se verán en mayor detalle.

3.2.1.1 Selección

Los algoritmos de selección serán los encargados de escoger qué individuos van a disponer de oportunidades de reproducirse y cuáles no. Puesto que se trata de imitar lo que ocurre en la naturaleza, se ha de otorgar un mayor número de oportunidades de reproducción a los individuos más aptos. Por lo tanto, la selección de un individuo estará relacionada con su valor de ajuste. Sin embargo, no se deben eliminar por completo las opciones de reproducción de los individuos menos aptos, pues en pocas generaciones la población se volvería homogénea.

3.2.1.1.1 Selección por ruleta

A cada uno de los individuos de la población se le asigna una parte proporcional a su ajuste de una ruleta, de tal forma que la suma de todos los porcentajes sea la unidad. Los mejores individuos recibirán una porción de la ruleta mayor que la recibida por los peores. Generalmente, la población está ordenada en base al ajuste, por lo que las porciones más grandes se encuentran al inicio de la ruleta. Para seleccionar un individuo basta con generar un número aleatorio del intervalo [0...1] y devolver el individuo situado en esa

posición de la ruleta. Esta posición se suele obtener recorriendo los individuos de la población y acumulando sus proporciones de ruleta hasta que la suma exceda el valor obtenido, presenta el inconveniente de que el peor individuo puede ser seleccionado más de una vez.

3.2.1.1.2 Selección por torneo

En este método, la selección consiste en escoger a los individuos genéticos en base a comparaciones directas entre sus genotipos. Existen dos versiones de selección mediante torneo, el torneo determinístico y el torneo probabilístico.

En la versión determinista se selecciona al azar un numero *p* de individuos, de entre los individuos seleccionados, se selecciona el más apto para pasarlo a la siguiente generación.

La versión probabilística se diferencia en el paso de selección del ganador del torneo, en vez de escoger siempre el mejor se genera un numero aleatorio del intervalo [0...1], si es mayor que un parámetro *p* se escoge el individuo más alto y en caso contrario el menos apto. Generalmente *p* toma los valores en el rango $0.5 < p \leq 1$.

Variando el número de individuos que participan en cada torneo se puede modificar la presión de selección. Cuando participan muchos individuos en cada torneo, la presión de selección es elevada y los peores individuos apenas tienen oportunidades de reproducción.

3.2.1.2 Cruce

Una vez seleccionados los individuos estos son recombinados para producir la descendencia que se insertará en la siguiente generación, su importancia para la transición entre generaciones es elevada puesto que las tasas de cruce con las que se suele trabajar rondan el 90%.

La idea principal del cruce se basa en que, si se toman dos individuos correctamente adaptados al medio y se obtiene una descendencia que comparta genes de ambos, existe la posibilidad de que los genes heredados sean precisamente los causantes de la bondad de los padres. Al compartir las características buenas de los individuos, la descendencia o al menos parte de ella, debería tener una bondad de los padres. Si el cruce no agrupa las mejores características en uno de los hijos y la descendencia tiene un peor ajuste que los padres, no significa que se esté dando un paso atrás, optando por una estrategia de cruce no destructiva garantizamos que pasen a la siguiente generación los mejores individuos. Si aún con un ajuste peor, se opta por insertar a la descendencia, y puesto que los genes de los padres continuarán en la población, aunque dispersos y posiblemente

modificados por la mutación, en posteriores cruces se podrán volver a obtener estos padres, recuperando así la bondad previamente pérdida.

3.2.1.2.1 Cruce de un punto

Una vez seleccionados dos individuos se cortan sus cromosomas por un punto seleccionado aleatoriamente para generar dos segmentos diferenciados en cada uno de ellos: la cabeza y la cola. Se intercambian las colas entre los dos individuos para generar los nuevos descendientes. De esta manera ambos descendientes heredan información genética de los padres (Ver figura 3.3), a este tipo de cruce suele llamarse SPX (Single Point Exchange).

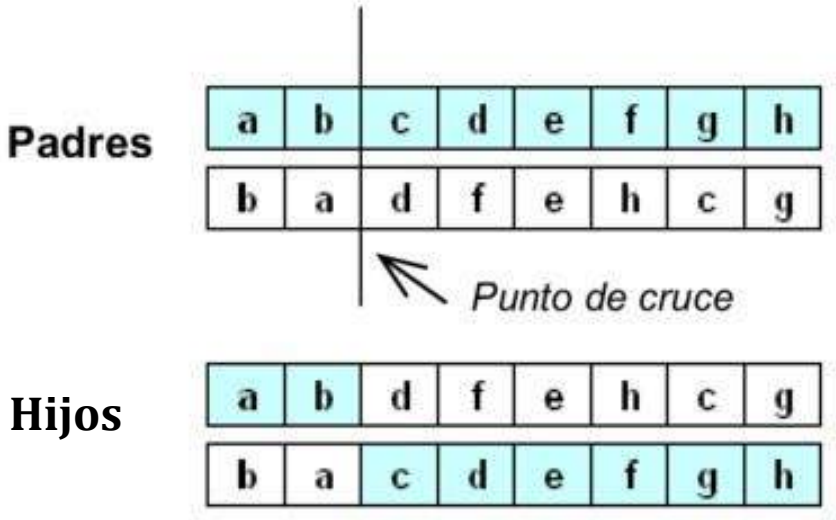


Figura 3.3: Cruce de un punto

3.2.1.2.2 Cruce de dos puntos

Se trata de una generalización del cruce de 1 punto, en vez de cortar por un único punto los cromosomas de los padres, como en el caso anterior, se realizan dos cortes. Deberá tenerse en cuenta que ninguno de estos puntos de corte coincida con el extremo de los cromosomas para garantizar que se originen tres segmentos. Para generar la descendencia se escoge el segmento central de uno de los padres y los segmentos laterales del otro padre (Ver figura 3.4), generalmente se refiere a este tipo de cruce como DPX (Double Point Crossover).

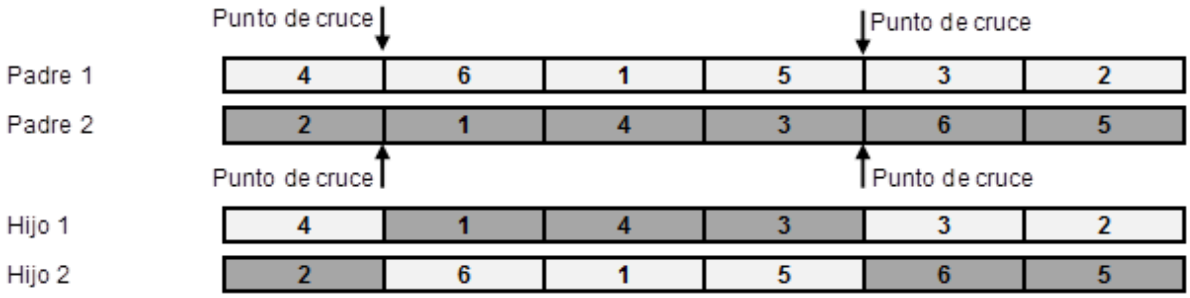


Figura 3.4: Cruce de dos puntos

3.2.1.2.3 Cruce uniforme

El cruce uniforme es una técnica completamente diferente de las vistas hasta el momento. Cada gen de la descendencia tiene las mismas probabilidades de pertenecer a uno u otro padre (Ver figura 3.5).

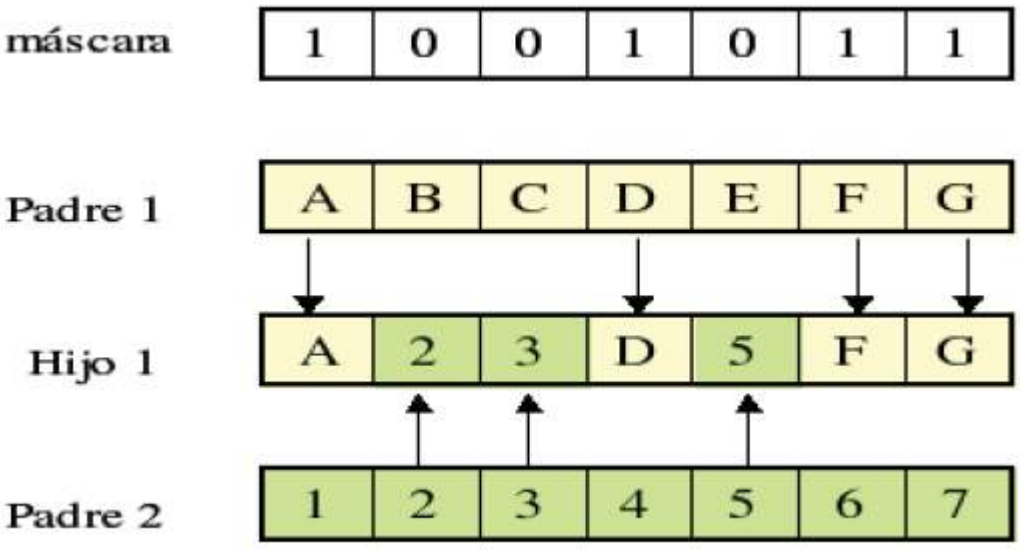


Figura 3.5: Cruce uniforme

La técnica implica la generación de una máscara de cruce con valores binarios, si en una de las posiciones de la máscara hay un 1, el gen situado en esta posición en uno de los

descendientes se copia del primer padre, si por el contrario hay un 0 el gen se copia del segundo padre. Para producir el segundo descendiente se intercambian los papeles de los padres, o bien se intercambian la interpretación de los unos y ceros de la máscara de cruce.

La máscara de cruce puede no permanecer fija durante todo el proceso evolutivo, se genera de manera aleatoria para cada cruce, se representa a este tipo de cruce son las siglas UPX (Uniform Point Crossover).

3.2.1.2.4 Cruce especifico de codificaciones no binarias

Si se emplean genotipos compuestos por valores enteros o reales pueden definirse otro tipo de operadores de cruce:

- Media: el gen de la descendencia toma el valor medio de los genes de los padres, tiene la desventaja de que únicamente se genera un descendiente en el cruce de dos padres.
- Media geometría: cada gen de la descendencia toma como valor la raíz cuadrada del producto de los padres.
- Extensión: se toma la diferencia existente entre los genes situados en las mismas posiciones de los padres y se suma al valor más alto o se resta del valor más bajo. Solventa el problema de generar un único descendiente.

3.2.1.3 Algoritmos de Reemplazo

Se trabaja con una sola población, sobre la que se realizan las selecciones e inserciones, deberá tenerse en cuenta que para insertar un nuevo individuo deberá de eliminarse previamente otro de la población, existen diferentes métodos de reemplazo:

- Aleatorio: el nuevo individuo se inserta en un lugar escogido de manera aleatoria en la población.
- Reemplazo de padres: se obtiene espacio para la nueva descendencia liberando el espacio ocupado por los padres.
- Reemplazo de similares: una vez obtenido el ajuste de la descendía se selecciona un grupo de individuos de la población con un ajuste similar, se reemplazan aleatoriamente los que sean necesarios.
- Reemplazo de los peores: de entre un porcentaje de los peores individuos de la población se seleccionan aleatoriamente los necesarios para dejar sitio a la descendencia.

3.2.1.4 Copia

Se trata de una estrategia de reproducción asexual, consiste en la copia de un individuo en la nueva generación. El porcentaje de copias de una generación a la siguiente es relativamente reducido, pues en caso contrario se corre el riesgo de una convergencia prematura de la población hacia ese individuo.

En un mejor caso se seleccionan dos individuos para el cruce y si este finalmente no tiene lugar, se insertan en la siguiente generación los individuos seleccionados.

3.2.1.5 Elitismo

El elitismo es un caso particular del operador de copia, consistente en copiar siempre al mejor, o en otro caso mejores individuos de una generación, en la generación siguiente. De esta manera se garantiza que el proceso de búsqueda nunca dará un paso atrás en cuanto a la calidad de la mejor solución obtenida, sino que un cambio en ésta siempre implicará una mejora. Una variación de este proceso consiste en copiar el mejor o mejores individuos de una generación en la siguiente, únicamente cuando tras el paso de una generación no se haya mejorado con los operadores de cruce o mutación la mejor solución de la generación actual.

3.2.1.6 Mutación

La mutación de un individuo provoca que alguno de sus genes, generalmente uno sólo, varíe su valor de forma aleatoria. Aunque se pueden seleccionar los individuos directamente de la población actual y mutarlos antes de introducirlos en la nueva población, la mutación se suele utilizar de manera conjunta con el operador de cruce. Primeramente, se seleccionan dos individuos de la población para realizar el cruce. Si el cruce tiene éxito entonces uno de los descendientes, o ambos, se muta con cierta probabilidad *Pm*. Se imita de esta manera el comportamiento que se da en la naturaleza, pues cuando se genera la descendencia siempre se produce algún tipo de error, por lo general sin mayor trascendencia, en el paso de la carga genética de padres a hijos.

La probabilidad de mutación es muy baja, generalmente menor al 1%, esto se debe sobre todo a que los individuos suelen tener un ajuste menor después de ser mutados, sin embargo, se realizan mutaciones para garantizar que ningún punto del espacio de búsqueda tenga una probabilidad nula de ser examinado.

La mutación más usual es el reemplazo aleatorio, este consiste en variar aleatoriamente un gen de un cromosoma, si se trabaja con codificaciones binarias, consistirá simplemente en negar un bit. También es posible realizar la mutación intercambiando los valores de dos alelos del cromosoma. Con otro tipo de codificaciones no binarias existen otras opciones:

- Incrementar o decrementar a un gen una pequeña cantidad generada aleatoriamente.
- Multiplicar un gen por un valor aleatorio próximo a 1.

3.3 Evaluación

Para que un Algoritmo Genético funcione correctamente, debe poseer un método que indique si los individuos de la población representan o no buenas soluciones al problema planteado, por lo tanto, para cada tipo de problema que se desee resolver deberá derivarse un nuevo método. De esto se encarga la función de evaluación, que establece una medida numérica de la bondad de una solución, esta medida recibe el nombre de ajuste. En la naturaleza, el ajuste (fitness) de un individuo puede considerarse como la probabilidad de que ese individuo sobreviva hasta la edad de reproducción y se reproduzca. Esta probabilidad deberá estar ponderada con el número de individuos de la población genética.

En el mundo de los Algoritmos Genéticos se empleará esta medición para controlar la aplicación de los operadores genéticos, es decir, permitirá controlar el número de selecciones, cruces, copias y mutaciones llevadas a cabo.

La principal aproximación consiste en crear explícitamente una medida de ajuste para cada individuo de la población. A cada uno de los individuos de la población se le asigna un valor de ajuste escalar por medio de un procedimiento de evaluación bien definido.

3.3.1 Fitness Puro

Es la medida de ajuste establecida en la terminología natural del propio problema, representado por la ecuación (3.1).

$$r(i,t) = \sum_{j=1}^{Nc} |S(i,j) - C(i,j)| \qquad (3.1)$$

Donde:

r(i,t): bondad del individuo *i* en la generación *t*

S(i,j): valor deseado para individuo *i* en el caso *j*

C(i,j): valor obtenido por el individuo *i* en el caso *j*

Nc: Número de casos

3.3.2 Fitness Estandarizado

Para solucionar los problemas de minimización o maximización se modifica el ajuste puro de acuerdo a la formula (3.2).

$$S(i,t) = \begin{cases} r(i,t) & minimización \\ rmax - r(i,t) & maximización \end{cases} \quad (3.2)$$

En el caso de minimización se emplea directamente la medida de fitness puro. Si el problema es de maximización se resta de una cota superior r_{max} del error el fitness puro. Por lo tanto, dentro de la generación *t*, un individuo *i* siempre será mejor que uno *j* si se verifica que *s(i,t) < s(j,t)*.

3.3.3 Fitness Ajustado

Este se obtiene aplicando la siguiente transformación al fitness estandarizado (3.3).

$$a(i,t) = \frac{1}{1+s(i,t)} \quad (3.3)$$

De esta manera el fitness ajustado tomará siempre valores del intervalo [0,1], cuando más se aproxime al fitness ajustado de un individuo a 1, mayor será su bondad.

3.3.4 Fitness Normalizado

El fitness normalizado indica la bondad de una solución con respecto al resto de soluciones representadas en la población, se obtiene mediante la siguiente ecuación: (3.4)

$$n(i,t) = \frac{a(i,t)}{\sum_{k=1}^{M} a(k,t)} \quad (3.4)$$

Al igual que el fitness ajustado, siempre tomará valores del intervalo [0,1], con mejores individuos cuanto más próximo esté a la unidad. Pero a diferencia de antes, un valor cercano a 1 no sólo indica que ese individuo represente una buena solución al problema, sino que además es una solución destacadamente mejor que las proporcionadas por el resto de la población. La suma de los valores de fitness normalizado de una población da siempre el valor de 1 [22].

4. IMPLEMENTACIÓN DEL ALGORITMO

El algoritmo implementado se utilizó para resolver el problema de búsqueda de motivos en secuencias de ADN, el cual es el algoritmo MDGA (Motif Discovery using Genetic Algorithm) [24].

4.1 Algoritmo MDGA

En este algoritmo se buscan motivos potenciales de un grupo de secuencias de ADN del sitio de inicio de la transcripción (TSS). Las operaciones genéticas como la mutación, y la cruza se realizan con la ayuda de la matriz de peso de posición generada a partir de un conjunto de secuencias emparejadas.

Se predicen motivos potenciales en secuencias promotoras mediante el uso de una función de puntuación total de aptitud y frecuencia de aparición. Además del algoritmo FMGA original, se presenta un nuevo parámetro en el cálculo de la puntuación total de aptitud, sobre la base de dos tipos de pares, purina y piridina. Se incluyen los pares de bases de ambigüedad IUPAC V, H, D, B para calcular sanciones del código de ambigüedad para penalizar la puntuación del motivo candidato y poder predecir mejores motivos con más eficiencia.

Se mantiene al hijo en la cruza para ser evaluado en la próxima generación y disminuir el riesgo de que un hijo sea eliminado, y este pueda ser un mejor motivo candidato. Para mantener el creciente número de motivos candidatos bajo control, se tiene una función de discursión que elimina comparativamente los motivos débiles de cada generación.

Los motivos de la primera generación con la misma longitud del motivo se crean aleatoriamente. Todos los motivos evolucionarán de la primera generación, después evolucionan a la generación *N*. El procesamiento previo de la primera generación es hacer que los motivos generados aleatoriamente sean más adecuados para la evolución. La mutación se realiza usando una matriz de peso, que reduce el tiempo de descubrimiento de posibles motivos candidatos (Ver figura 4.1).

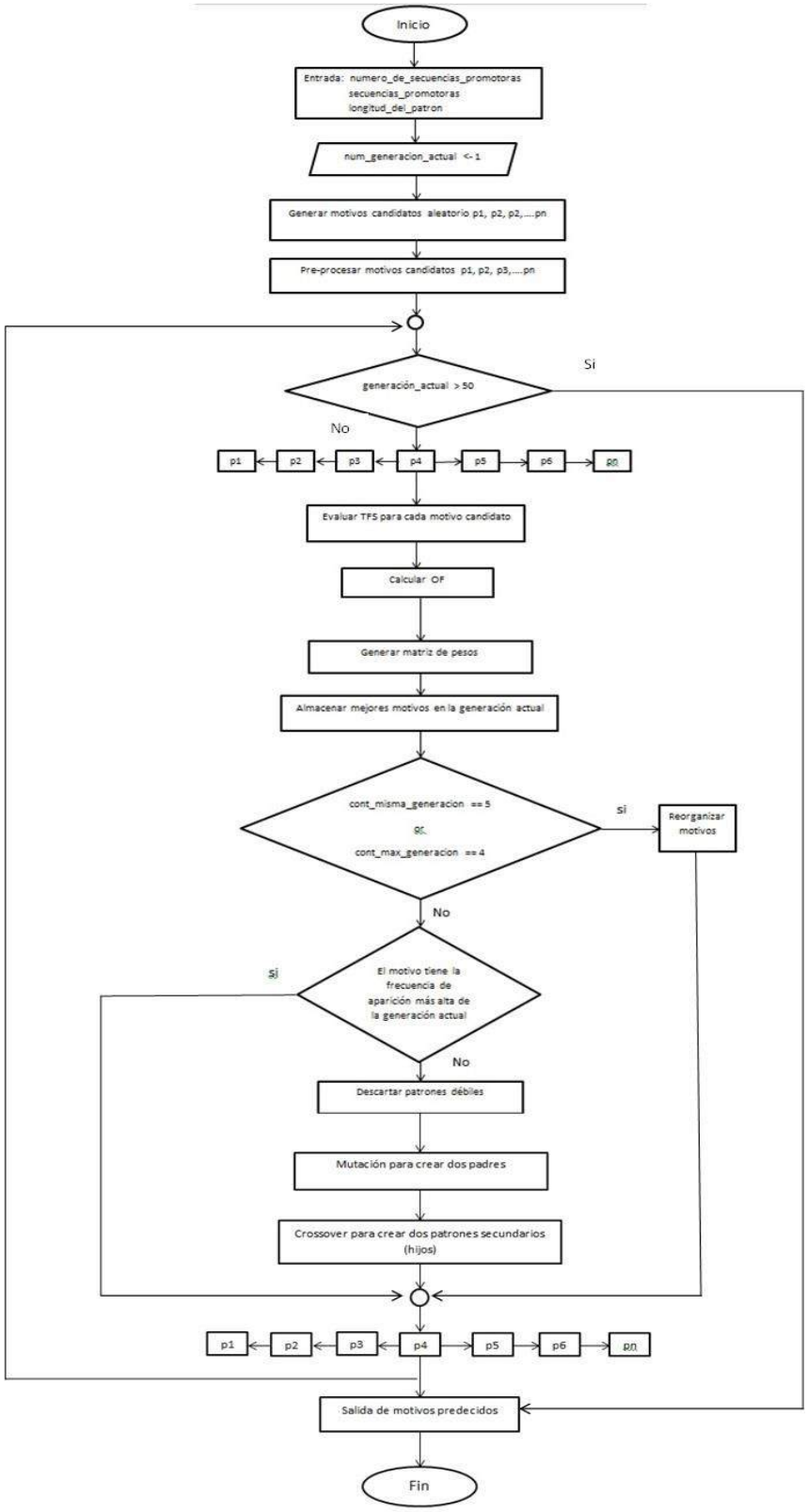


Figura 4.1: Arquitectura MDGA.

A continuación, se explicará a detalle el algoritmo implementado.

Donde, S es la secuencia evaluada y P es el motivo a evaluar.

A. Función de puntuación de aptitud física

La función de aptitud para un motivo candidato en una sola secuencia es definida como:

$$FS(Smj, Pn) = \sum_{i=0}^{k} match(Smji, Pni)/k \tag{4.1}$$

Donde,

$$match(Smji, Pni) = \begin{cases} 1 & if Smij = Pni \\ 0 & if\ Smij\ != Pni \end{cases} \tag{4.2}$$

m: índice de secuencias promotoras

i: posición en la secuencia

j: número de regiones emparejadas en la secuencia

k: longitud del motivo

n: índice de motivos

Además de los pares de bases A, T, G, C también se utilizan los pares de bases de ambigüedad M, R, W, S, Y, K, V, H, B, D, en el cálculo de la puntuación de aptitud física, también son considerados los pares de bases purina y piridina, ya que tienden a convertirse en una ambigüedad frecuente y están representados por R(A, G) e Y(T, C). Y se define la condición de emparejamiento (4.3).

$$match(Smij, Pni) = f(x) = \begin{cases} 1 & if\ Smij = Pni\ for\ Pni\ \in (A, T, G, C) \\ 0.5 & if\ Smij = Pni\ for\ \in (R, Y) \\ 0.2 & if\ Smij = Pni\ for\ \in (M, N, S, K) \\ 0.1 & if\ Smij = Pni\ for\ \in (B, V, H, D) \\ 0 & if\ Smi\ != Pni \end{cases} \tag{4.3}$$

B. Puntuación física total

La puntuación de aptitud total de un motivo es la suma acumulativa de la puntuación de aptitud física que tenga la mejor coincidencia en todas las secuencias promotoras. Representa la sección del motivo en una generación en particular (4.4).

$$TFS(S, Pn) = \sum_{m=1}^{L} FS(Sm, Pn) \tag{4.4}$$

A continuación, se muestra el cálculo de la puntuación física total, teniendo como motivo candidato a P1 en seis secuencias promotoras.

P1: GATGAA

S1: AGCTCTAGAGACAG	⟶	3/6	=	0.5
S2: TTAAGGCCAACGTT	⟶	3/6	=	0.5
S3: GCCTAAGGCACTGG	⟶	3/6	=	0.5
S4: CATTAGGATGAACA	⟶	6/6	=	1
S5: TCCCATGGCCATAT	⟶	3/6	=	0.5
S6: GAAGATAAGACTAC	⟶	4/6 = 0.66		

TFS (S, P1) = 4.16

C. Frecuencia de ocurrencia

La frecuencia de ocurrencia (OF) de un motivo dado se define como el número de secuencias en un conjunto de datos dado, en el cual, el motivo candidato coincide completamente con una subsecuencia de la secuencia promotora.

En el siguiente ejemplo se muestra el cálculo de la frecuencia de ocurrencia.

P1: GATGAA

S1: AGCTCTAGAGACAG	⟶	3/6	=	0.5
S2: TTAAGGCCAACGTT	⟶	3/6	=	0.5
S3: GCCTAAGGCACTGG	⟶	3/6	=	0.5
S4: CATTAGGATGAACA	⟶	6/6	=	1
S5: TCCCATGGCCATAT	⟶	3/6	=	0.5
S6: GAAGATAAGACTAC	⟶	4/6 = 0.66		

OF (P1) = 1

D. Operaciones

Las operaciones importantes en MDGA son preprocesamiento, mutación, cruza, reordenamiento y discursión que pueden reducir el tiempo computacional del algoritmo y predecir mejores motivos.

- Preprocesamiento

Después de crear un motivo aleatorio al comienzo del proceso, este pasa por un mecanismo de preprocesamiento para convertirlo en la secuencia de pares de bases del código de ambigüedad. Después del preprocesamiento, los motivos candidatos estarán bastante relacionados con el promotor de secuencias y eventualmente se convierten en mejores motivos candidatos con alto potencial en generaciones posteriores.

Al comienzo del preprocesamiento, se genera el peso de la posición de la matriz de cada motivo. La matriz de peso se genera a partir del mejor patrón coincidente de cada secuencia cruzada a un motivo candidato. Considerando los 5 mejores patrones combinados de un motivo candidato P1, el valor de una sola celda en peso, la matriz se genera como la proporción de apariciones de las bases correspondientes y el número de motivos emparejados.

A continuación, se muestra la creación de la matriz de pesos en la Tabla 4.1.

G C T C T A
G G C C A A
G C C T A A
G A T G A A
G A A G A T

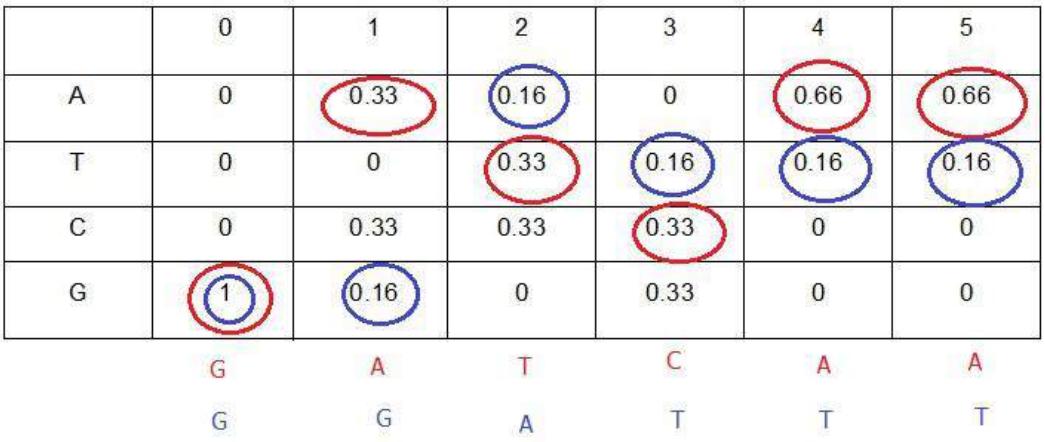
	0	1	2	3	4	5
A	0	0.33	0.16	0	0.66	0.66
T	0	0	0.33	0.16	0.16	0.16
C	0	0.33	0.33	0.33	0	0
G	1	0.16	0	0.33	0	0

Tabla 4.1: Matriz de pesos

- Mutación

El objetivo de la mutación es crear dos motivos parentales a partir de un motivo candidato para acelerar el proceso de búsqueda del motivo potencial. Es útil en la operación de cruza para crear dos hijos motivos para una mayor evolución. La mutación se realiza en función de la matriz de peso de un motivo candidato. Primero, se mantiene el par de bases con valor de 1 en una columna sin cambios y el resto de los pares de bases se cambian aleatoriamente. Para el primer padre se usa el par de bases con el valor máximo en una columna. Para el segundo padre, se utiliza el par de bases hasta el segundo valor más alto en la columna.

A continuación, se muestra la forma en que se seleccionan los padres a través de la mutación.



.Figura 4.2: Creación de dos padres.

Cada columna representa una posición del motivo candidato, para el primer padre se toma el valor más alto de cada columna (encerrados en rojo) y se coloca la letra de la fila correspondiente. El valor del segundo padre se obtiene tomando el segundo valor más alto de cada columna (encerrados en azul) y de igual forma se toma la letra de la fila que le corresponde (Ver Figura 4.2).

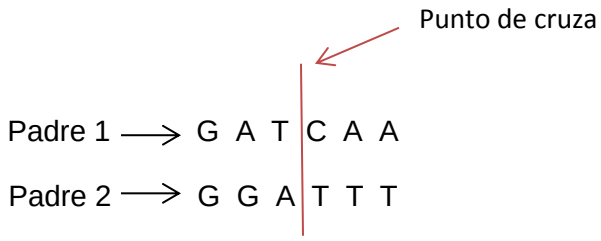
Padre 1 ⟶ G A T C A A

Padre 2 ⟶ G G A T T T

- Cruza

Se crean dos motivos secundarios a partir de dos padres motivos, producidos en la mutación. Se implementa la cruza de un solo punto. Primero, se corta a los dos padres en la posición media. Entonces se une el lado izquierdo del primer padre con el lado derecho del segundo padre para crear el primer motivo hijo. Después se une el lado derecho del primer padre con el lado izquierdo del segundo padre para crear el segundo motivo hijo.

En el siguiente ejemplo se muestra como se obtienen los hijos a partir de los padres mutados.



Hijo 1 ⟶ G A T T T T

Hijo 2 ⟶ G G A C A A

- Reordenamiento

Los motivos predichos o la frecuencia de ocurrencia máxima de una generación en particular pueden permanecer sin cambios para más de *x* generaciones. En ese caso se reorganizan los motivos candidatos al convertir los códigos de ambigüedad en A, T, C y G según la matriz de pesos. Se reemplaza el código de ambigüedad con el nucleótido que tiene la puntuación más alta en la columna correspondiente de la matriz de pesos. Si el

código de ambigüedad no existe en el motivo candidato, entonces se modifica el par de bases en las posiciones respectivas según la matriz de pesos.

- Discursión

Mientras se mantiene a los hijos producidos por la cruza, el número de motivos candidatos se seguirá multiplicando en cada generación. Entonces, si no se tiene una medida para reducir el número de motivos candidatos de alguna manera, el tiempo de cálculo de cada generación aumentará exponencialmente.

Para minimizar la complejidad de tiempo, después de mutar tantos motivos como sea posible, se introducirá una función asociada por un factor de discursión. Inicialmente, la función descartará una pequeña cantidad de motivos candidatos que tienen un valor TFS bajo y menos ocurrencia de frecuencia (OF) de cada generación.

Con el aumento de los motivos candidatos en cada generación, el factor de discursión se ajustará para mantener baja la cantidad de motivos, y que aumente más de cuatro veces que el número inicial de motivos candidatos.

Se evitará descartar cualquier motivo candidato que tenga más posibilidades de evolución a través de la mutación y la cruza.

5. PRUEBAS Y RESULTADOS

En este capítulo se muestra el procedimiento y los datos utilizados para la búsqueda de motivos en secuencias de ADN.

A continuación, se presenta un ejemplo de prueba para el algoritmo implementado con la finalidad de verificar la funcionalidad y efectividad del mismo.

La base de datos utilizada para la ejemplificación del algoritmo es GenBank [23].

Las especificaciones de la computadora en donde se realizaron las pruebas:

- Procesador AMD A8-4555M
- Memoria RAM 8 GB
- CPU a 1.60 GHz

El software implementado se desarrolló en lenguaje C y se compilo en GCC.

En esta prueba se toman como ejemplo los siguientes parámetros:

Tamaño de secuencias: 40

Numero de secuencias: 30

Tamaño del motivo: 15

Motivo candidato: CCTAGATCTGGTTAC

En donde el tamaño de secuencias es el largo que cada una de las secuencias tiene, numero de secuencias es la población inicial donde se tendrá el espacio de búsqueda del motivo candidato, el tamaño del motivo es el largo del motivo candidato y el tamaño que tendrán los motivos predichos, y el motivo candidato es aquel que se evaluará en las secuencias para obtener a los mejores motivos, además se tiene un numero de iteraciones de 50.

El programa muestra los mejores motivos, es decir, aquellos motivos que tienen una mayor coincidencia con el motivo candidato y no se repiten (Ver figura 5.1).

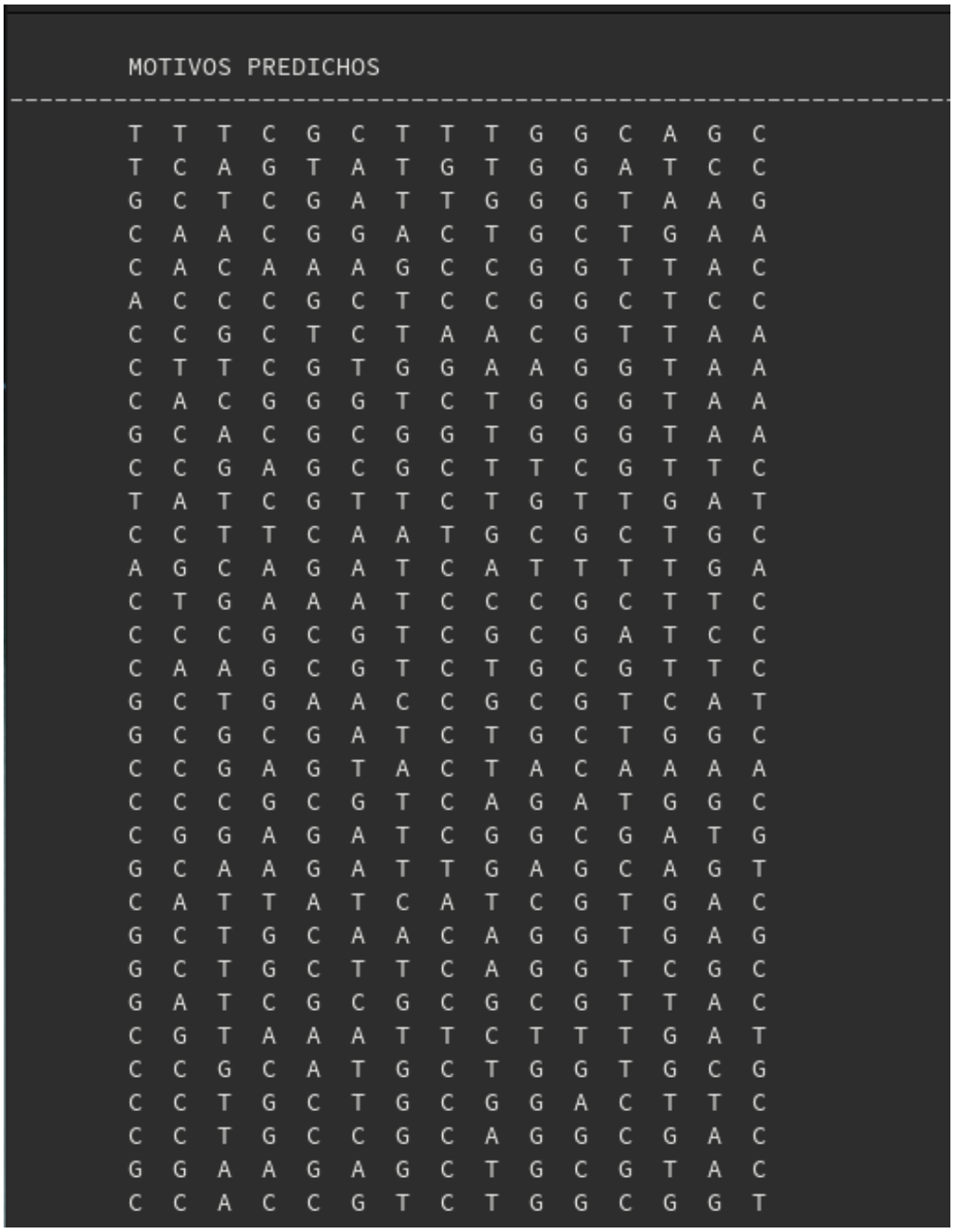


Figura 5.1: Mejores motivos obtenidos.

Como se muestra en la imagen anterior, se obtienen los motivos con mayor coincidencia al motivo candidato, aunque no se encontró completamente, se obtiene una ocurrencia de 10 posiciones, con lo cual se puede notar que busca los mejores resultados.

Se utilizaron tres grupos diferentes para probar el algoritmo, el primer grupo consta de 210 secuencias promotoras y de tres tamaños de motivos candidatos diferentes: 15, 8 y 22, cada tamaño con dos motivos distintos. Este primer grupo fue tomado de GenBank, al tener tres tamaños diferentes se aprecian las variaciones que se pueden obtener ya sea solo con cambiar el tamaño del motivo. Como se puede apreciar en la Tabla 5.1 mientras el tamaño del motivo sea de menor tamaño, el TFS obtiene mejores resultados, al tener un tamaño de 8 el TFS es superior al 50% del total de las secuencias promotoras totales; al contrario del tamaño 22 donde el TFS es apenas un 42.33% compatible con el total de secuencias.

TAMAÑO DEL MOTIVO CANDIDATO	**MOTIVO CANDIDATO**	**TFS**	**OCURRENCIA**
15	CCTAGATCTGGTTAC	105	10
	ATCCRGATBATTGAG	107	10
8	CCATSMGT	113.25	5
	ATGGCATT	132.25	7
22	AATCCGGACCAGGMDTACCGAT	93.81	13
	BCCGRCTAGGCATSDTTAGGAC	88.90	11

Tabla 5.1: Motivos predichos grupo uno.

TAMAÑO DEL MOTIVO CANDIDATO	MOTIVO CANDIDATO	TFS	OCURRENCIA
15	CCTAGATCTGGTTAC	26.06	9
	ATCCRGATBATTGAG	26.66	9
8	CCATSMGT	29.37	5
	ATGGCATT	32.50	6
22	AATCCGGACCAGGMDTACCGAT	23.47	11
	BCCGRCTAGGCATSDTTAGGAC	22.34	10

Tabla 5.2: Motivos predichos grupo dos.

En el segundo grupo se tiene un total de 55 secuencias promotoras además de tres los tamaños de motivos candidatos diferentes: 15, 8 y 22, este grupo también fue tomado de GenBank. En este grupo se puede apreciar que el TFS no varía tanto como el ejemplo anterior, como se muestra en la Tabla 5.2, pero de igual manera al tener un tamaño de motivo menor se obtienen mejores resultados, en el tamaño 8 se tiene un TFS de 59.09% del total de secuencias promotoras, este resultado es superior a la media, lo que indica que este grupo obtiene mejores resultados que el anterior.

El grupo tres está conformado de 18 secuencias promotoras, además de tres tamaños de motivos candidatos diferentes: 15, 8 y 22. En este caso se puede notar una mayor variación de resultados como se muestra en la Tabla 5.3, ya que el mejor TFS es de 58.3% del total de las secuencias promotoras y el peor resultado es 31.8% del total con el tamaño del motivo de 22, siendo el de mejor resultado el tamaño de motivo de 8.

TAMAÑO DEL MOTIVO CANDIDATO	MOTIVO CANDIDATO	TFS	OCURRENCIA
15	CCTAGATCTGGTTAC	7.46	8
	ATCCRGATBATTGAG	7.05	6
8	CCATSMGT	9.90	5
	ATGGCATT	10.50	6
22	AATCCGGACCAGGMDTACCGAT	5.74	10
	BCCGRCTAGGCATSDTTAGGAC	5.78	8

Tabla 5.3: Motivos predichos grupo tres.

Como se puede notar en las tablas anteriores los datos siempre son muy cambiantes, y mientras el motivo candidato sea menor se obtendrá una mayor ocurrencia, así como el tener una mayor cantidad de secuencias promotoras, ya que esto le permite al algoritmo una mayor cantidad de búsquedas, esto también es debido a que el motivo candidato se reorganiza determinado número de iteraciones, evitando que los mismos resultados se repitan en generaciones posteriores.

6. CONCLUSIONES

En este capítulo se presentan las conclusiones de la investigación realizada.

Las pruebas se realizaron con diferentes tamaños de bases de datos, por lo cual se pudo percibir que entre menor sea el tamaño del motivo candidato se obtienen mejores resultados, ya que se tiene una mayor compatibilidad entre el motivo a evaluar y las secuencias promotoras.

Tomando en cuenta los resultados descritos en el capítulo anterior, los resultados son positivos, ya que al tener una poca cantidad de secuencias se logra obtener los mejores motivos, y notando que ninguno de estos se repite. Ya que en esto influye el número de generaciones utilizadas, ya que esa cantidad es la que es con la que el algoritmo puede obtener las siguientes poblaciones y que estas no sean repetitivas.

Con esto se puede determinar que el algoritmo MDGA da buenos resultados en la búsqueda de motivos en secuencias de ADN.

GLOSARIO

k-mers. Son subsecuencias de longitud *k* contenida dentro de una secuencia biológica.

Motivos borrosos. Son aquellos motivos que son extensos y poco precisos.

PWM. (Position-specific Weight Matrices) por sus siglas en inglés Matrices de peso específicas de la posición.

Genotipo. Conjunto de la información genética almacenada en el ADN de un organismo particular, cuya totalidad en términos de especie compone el genoma.

Fenotipo. Se refiere a las características físicas observables de un organismo, producto de la expresión o manifestación de la información genética contenida en el genotipo, en concordancia con las condiciones del ambiente determinado en el que hace vida el organismo.

Alelo. Un alelo es cada una de las dos o más versiones de un gen. Un individuo hereda dos alelos para cada gen, uno del padre y el otro de la madre. Los alelos se encuentran en la misma posición dentro de los cromosomas homólogos.

Secuencias promotoras. Son secuencias de ADN necesarias para convertir un gen en activado o desactivado.

PMP. Problemas de búsqueda de motivos.

BIBLIOGRAFÍA

[1] Piro, O. Breve historia del ADN, su estructura y función. Universidad Nacional de la Plata.

[2] Copelli, S. (2010). *Genética: desde la herencia a la manipulación de los genes*. Buenos Aires. Fundación de Historia Natural Félix de Azara.

[3] Illana, J. (2014). Biología molecular y estructura del ADN. *Real Sociedad Española de Química, Vol. 110* (3), 234-240.

[4] Hashim, F., Mabrouk, M., Al-Atabany, W. (2018). Review of Different Sequence Motif Finding Algorithms. *Avicenna Journal of Medical Biotechnology, Vol. 11* (2), 135.

[5] Chauhan, R., Agarwal, P. (2012). A Review: Applying Genetic Algorithms for Motif Discovery. Int. J. *Computer Technology & Applications, Vol 3* (4), 1510-1515.

[6] Rombauts, S., Dehais, P., Van Montagu, M., Rouze, P. (1999) PlantCARE, a plant cis acting regulatory element database. *Nucleic Acids* Res. 295–296.

[7] NA. Motivos, Patrones y Perfiles. [PDF File]. 6-9.

[8] Hashim, F., Mabrouk, M., y Al-Atabany, W. (2019). Review of Different Sequence Motif Finding Algorithms*. Avicenna Journal of Medical Biotechnology*. 11 (2). 1-2.

[9] Bailey, T. (2011) DREME: motif Discovery in transcription factor ChIP-seq data. *Bioinformatics*. 27 (12). 1653-1659.

[10] Van Helden, J., Andre, B., Collado-Vides, J. (1998). Extracting regulatory sites from the upstream region of yeast genes by computational analysis of oligonucleotide frequencies. *J. Mol. Biol*., 281, 827–42.

[11] Alexei A., Minoru, S.H. (2009). Exahustive Search for Over-represented DNA Sequence Motifs with CisFinder. *DNA Research*. 16, 261-273.

[12] Sharov AA, Ko SH. Exhaustive search for over-represented DNA sequence motifs with CisFinder. *DNA Res* 2009;16(5):261-273.

[13] Pavesi, G., Mereghetti, P., Mauri, G. y Pesole, G. (2004). Weeder Web: Discovery of transcription factor binding sites in a set of sequences from co-regulated genes. *Nucleic Acids Research*. 32.

[14] Liang, S. (2003). cWINNOWER Algorithm for Finding Fuzzy DNA Motifs. *Proceedings of the Computational Systems Bioinformatics*.

[15] Kilpatrick, AM., Ward, B., Aitken, S. (2014) Stochastic EM-based TFBS motif discovery with MITSU. *Bioinformatics*. 30(12):i310–i318.

[16] Siebert, M., Soding, J., (2016) Bayesian Markov models consistently outperform PWMs at predicting motifs in nucleotide sequences. *Nucleic Acids Res.* 44(13).

[17] Perez, F. (2019). El enfoque probabilístico en Inteligencia Artificial. *Grupo de Estudios en Seguridad Nacional (GESI)*. 2340-8421.

[18] Bailey, T., Elkan, C. The MEME Suite. California. Recuperado de http://web.mit.edu/meme_v4.11.4/share/doc/meme.html

[19] German, S. (1984). Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 6 (6): 721-741.

[20] Xing, E., Wu, W., Jordan, M., Karp, R. (2004). LOGOS: A Modular Bayesian Model for de Novo Motif Detection. *Journal of Bioinformatics and Computational Biology.* 2 (1). 127-154.

[21] Garduño, R. (2018). Algoritmos genéticos. Conogasi: Aprender es conocer. Recuperado de: http://conogasi.org/articulos/algoritmos-geneticos/.

[22] Gestal, M., Rivero, D., Rabuñal, J., Dorado, J., Pazo, A. (2010). Introducción a los Algoritmos Genéticos y la Programación Genética. Madrid, España: Consorcio Editorial Galego.

[23] Varios. https://www.ncbi.nlm.nih.gov/genbank/

[24] Muttakin, A., Huq, M. (2017). Motif Discovery in Unaligned DNA Sequences Using Genetic Algorithm. 4th International Conference on Advances in Electrical Engineering (ICAEE). 725-730.