Universidad Autónoma De Coahuila

Facultad de Sistemas

Ingeniería en Sistemas Computacionales

Análisis y Modelación de Sistemas

Algoritmo Genético

Orlando Raúl García Vélez

30 de abril 2024

Conceptos básicos

Algoritmo: Es un conjunto de reglas que hay que seguir para realizar alguna tarea o un proceso. En informática esas reglas se le indican al ordenador para que ejecute esa tarea.

Metaheurística: Es una técnica, método o procedimiento inteligente de realizar una tarea que no es producto de un riguroso análisis formal, sino de conocimiento experto sobre la tarea.

Algoritmo genético: Es una técnica de programación inspirada en la reproducción de los seres vivos y que imita a la evolución biológica como estrategia para resolver problemas de optimización.

Bioinformática: Es una subdisciplina científica que implica el uso de ciencias informáticas para recopilar, almacenar y analizar y diseminar datos e información biológicos, como secuencias de ADN y aminoácidos o anotaciones sobre esas secuencias.

Introducción:

En el campo de la bioinformática, el algoritmo genético se basa en una herramienta inspirada en la evolución biológica.

La comparación de secuencias es una de las actividades fundamentales en el análisis bioinformático.

El alineamiento de secuencias es el proceso en el cual diferentes secuencias son comparadas mediante la búsqueda de patrones de caracteres comunes.

Propósito del proyecto:

El propósito de este proyecto consiste en encontrar un alineamiento óptimo o muy cercano para secuencias genéticas.

Desarrollar un algoritmo el cual sea capaz de alinear secuencias genéticas, siguiendo los pasos correspondientes y al pie de la letra de una un MSA (Alineamiento Múltiple de Secuencias).

Fases de un algoritmo genético

Población inicial:

El proceso comienza con la creación de una población inicial. Esta población está compuesta por un conjunto de individuos/secuencias, y cada uno representa una solución potencial al problema. Los individuos son caracterizados por un conjunto de parámetros conocidos como genes, que se unen para formar un cromosoma.

```
#Matriz inicial de nucleótidos
matriz_inicial = [
    'TGGTTGCGTTGCTTCTTCTGGCCATTTCGCATCTGTATAAC',
    'ATCAAAGATGAAATCTTCACCAAATGGCGCATTTTCTTCGATTTTATC',
    'TTGATTCTTTAGCATCTGGTTTAAATCCAACTACCACGGTTTG']
```

Evaluación (Fitness/Score):

Se evalúa cada individuo de la población utilizando una función de evaluación.

Esta función determina qué tan “buena” o “mala” es la solución que el individuo representa.

Se pueden usar muchos sistemas de evaluación tanto simples como muy complejos.

El score/fitness se calcula de esta manera:

. -Si se encuentra un gap con otro gap se le resta 1 punto.

. -Si se encuentra un gap y un carácter (no gap) se le restan 2 puntos.

. -Si son 2 caracteres iguales (no gap) se le suman 5 puntos.

. -Si son 2 caracteres no iguales se le restan 2 puntos.

Todo esto, se evalúa comparando caracteres columna por columna.

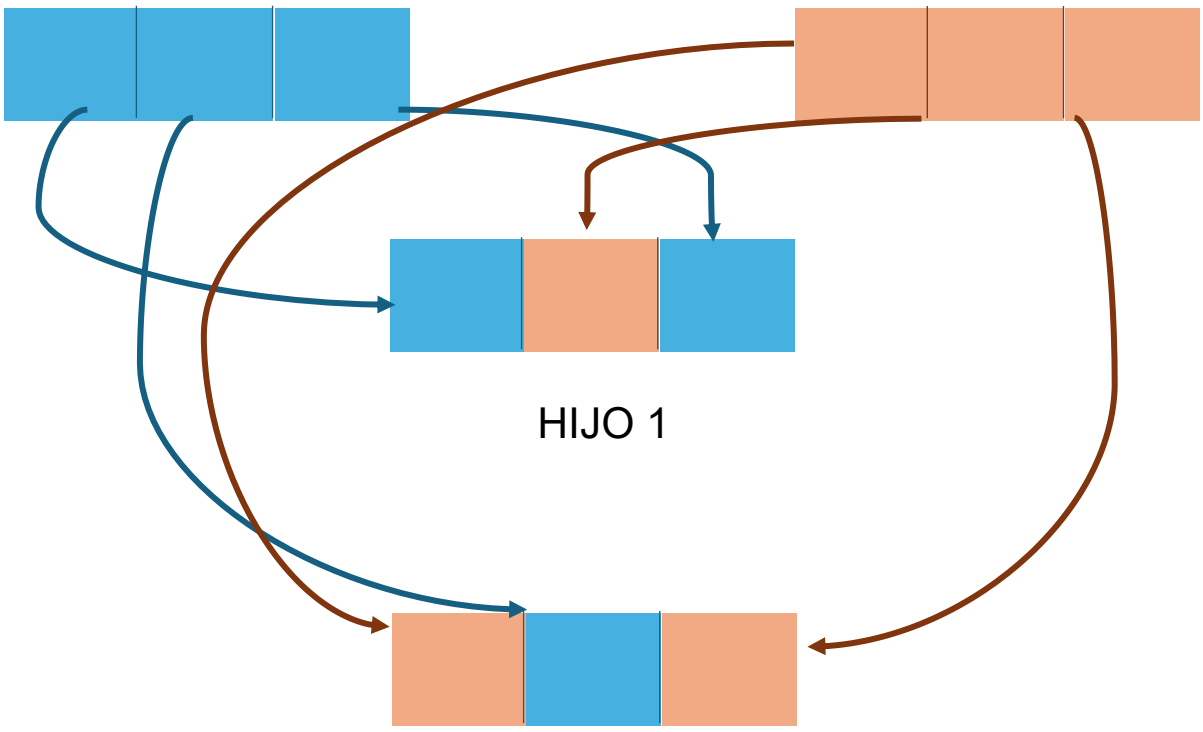
Selección de padres:

Basándose en la función de evaluación, se seleccionan los individuos más aptos. Estos individuos tienen mayor probabilidad de ser elegidos para pasar sus genes a la siguiente generación, también existen muchos métodos, en este algoritmo usaremos uno el cual funciona de esta manera, se seleccionan a las secuencias/individuos que tuvieron una evaluación mayor a los demás, para esto se tiene que ordenar después de la evaluación realizada:

PADRE MADRE

Cruza (Creación de los hijos):

Se toman pares de individuos seleccionados y se combinan sus cromosomas para producir descendencia. Este proceso simula la reproducción sexual y permite la mezcla de material genético.



La cruza que se usaremos se describe de esta forma:
. - Los padres se dividen en 3 partes por el tamaño de su longitud.

. - Para crear al primer hijo se seleccionan las dos partes exteriores del padre y la parte central de la madre.
. - Para el hijo 2 se seleccionan las partes restantes, es decir, los 2 extremos de la madre y la parte central del padre.

Mutación:

A los individuos de la nueva generación (los hijos) se les puede aplicar mutaciones con baja probabilidad. Esto introduce variabilidad en la población y simula las mutaciones aleatorias en la naturaleza.

Esta mutación se trata de agregar GAPS de manera aleatoria a esa secuencia, por ejemplo:

Secuencia sin gaps: **ATATTGCTACGTATATCAT**

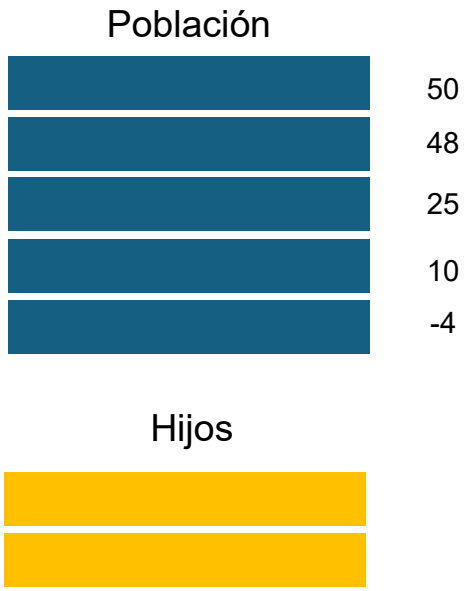
Secuencia con gaps: **ATATT-GCTAC-GTATA-TCAT**

La cantidad de gaps utilizadas en este proyecto es de 3 gaps.

Reemplazo:

La nueva generación de individuos reemplaza a la generación anterior (los individuos con menor evaluación se eliminan y se colocan los hijos creados y mutados), y el proceso se repite.

En cada generación, la población debería acercarse más a la solución óptima.



Nueva población

Repetición:

En esta parte del algoritmo se repite, todo el proceso o los pasos que se siguieron, desde la evaluación, con los nuevos individuos en la población se vuelve a evaluar, se seleccionan a los mejores, se dividen, se crean los hijos, se sustituyen por los menos aptos y se vuelve a empezar, una y otra vez, por un numero definido de generaciones.

El número de generaciones puede varían, pueden ser tan pocas como 100 o se pueden llegar a extender hasta 1,000 o 10,000 generaciones para encontrar un alineamiento.

Pruebas y ejecuciones

Para este proyecto se empezaron con pruebas en palabras que tuvieran muchas variantes de vocales y consonantes, para que fuera más fácil encontrar las similitudes y los cambios que se hacían. Se adjuntarán capturas de pantalla de las ejecuciones con palabras (pruebas), al final se mostrará la ejecución final con secuencias reales.

Para la evaluación se usaron las siguientes palabras:

. -Sincero

. -Cigarro

. -Cargador

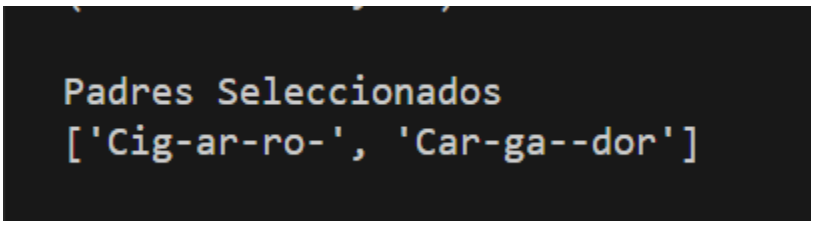
Ya que se nota a simple vista las letras que comparten, como se mencionó anteriormente usamos una forma de evaluar:

```
Nueva Poblacion
Cig-ar-ro-
Sincer-o--

Generación 1
Secuencias ordenadas por calificacion
('Cig-ar-ro-', 47)
('Car-ga--dor', 42)
('Sincer-o--', 42)
```

El proceso de evaluación consiste en seleccionar una secuencia, compararla con las demás, luego pasar a la siguiente secuencia y repetir este proceso con cada secuencia de la población

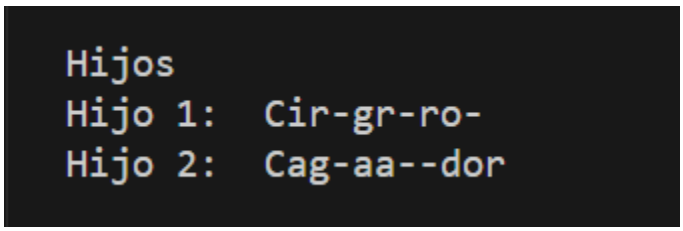
Selección de padres:



Tomando en cuenta las evaluaciones, nos damos cuenta de que esas palabras son las mejores, así que esas se toman como “padres” y de ahí se hace la función de partición y la cruza.

Cruza y partición:

Como se planteó la partición se haría en 3 partes **sin tomar en cuenta los GAPS.**



Nos damos cuenta de que la partición está hecha correctamente, por ejemplo;

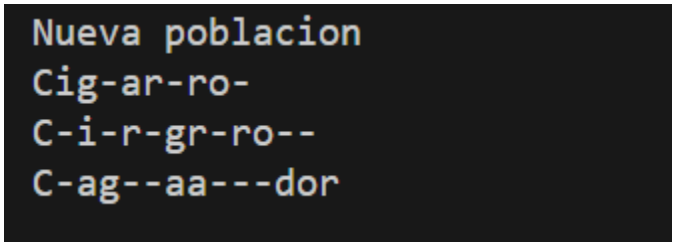
El primer hijo, en color amarillo pertenece al padre y la parte roja a la madre:

Cir-gr-ro-

Para el hijo 2 tenemos los mismos colores, pero con las partes restantes:

Cag-aa—dor

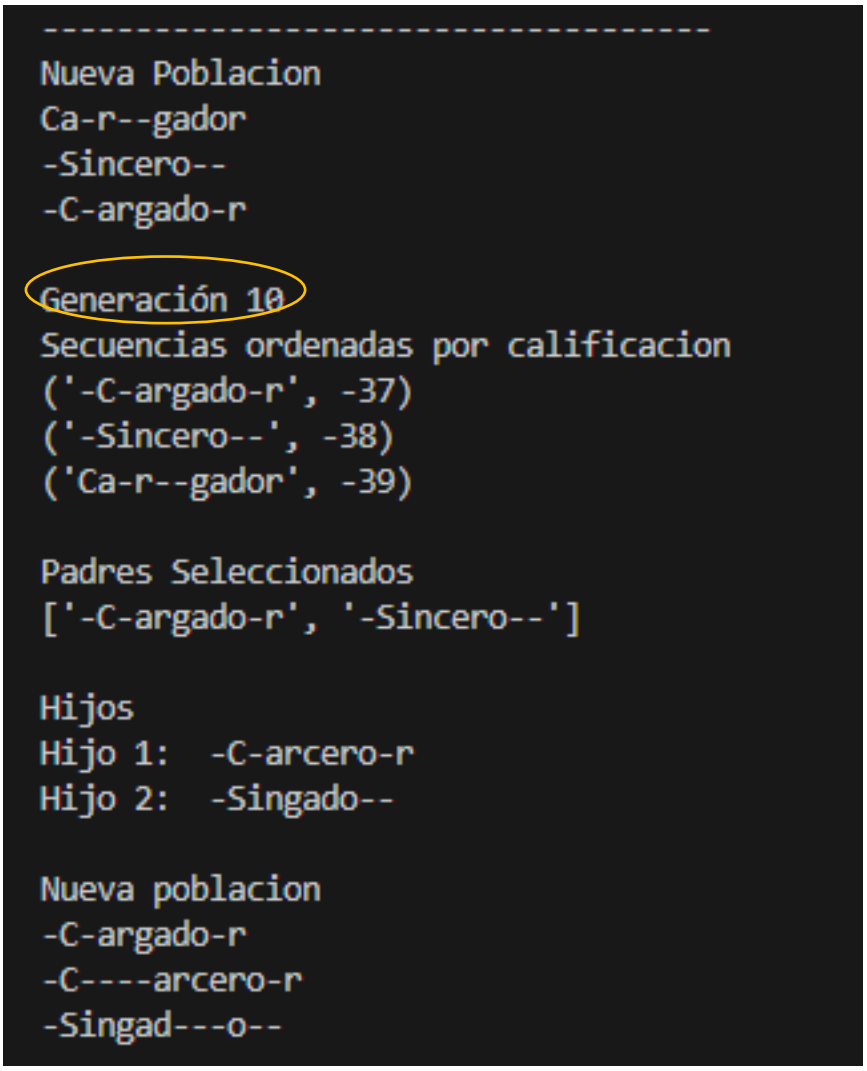
Sustitución:



Nos damos cuenta y comparándolo con la primera imagen que aparece la población, que se han sustituido de manera correcta.
Se han seleccionado los menos evaluados y se han colocado los hijos, en este caso que son pruebas y solamente tenemos 3 individuos en la población se eliminan, aunque se haya elegido 1 como padre.

Repetición:

Como se ha mencionado antes, se realiza un ciclo ajustando el número de generaciones que deseamos, por ejemplo, en este caso se han dado 10 generaciones.



Ejecución del código final:

. -2,000 generaciones.

Secuencias iniciales:

```
#Matriz inicial de nucleótidos
matriz_inicial = [
    'TGGTTGCGTTGCTTCTTCTGGCCATTTCGCATCTGTATAAC',
    'ATCAAAGATGAAATCTTCACCAAATGGCGCATTTTCTTCGATTTTATC',
    'TTGATTCTTTAGCATCTGGTTTAAATCCAACTACCACGGTTTG']
```

Ejecución:

```
Nueva Poblacion
TGGTTGCGT-TGCTTCTTCTGGCCATTTCGCA-TCTGTAT-AAC
TGG-TTGCGTT-GCTTCTTCTGGCCA-TTTCGCATCTGTATAAC
TGGTTGCGTTGCTTCT-TCTGGCCA-TTTCGCATCTGTATAA-C
ATCAAAGATGAAATCTT-CACCAAATGGCG-CATTTTCTTCGATTTTAT-C
-ATCAAAGATGAAATCTTCACCAAATGG-CGCATTTTCTT-CGATTTTATC
TGGTTGCGTTG--CTTCTTCTGGCCATTTCGCA-TCTGTATAAC
TGGTTGCGT-T-GCTTCTTCTGGCCATTTCGCATCTGT-ATAAC
TGGTTGCGTTGCTTCTT-CTGGCCAT-TTCG-CATCTGTATAAC
TGGTTGCG-TTGCTTCTTCTGGCCATTT-CGCATCTGTA-TAAC
TTGATTCTTTAGCATCTGGTTTAAATCCAAC-TACCACGG-T-TTG
```

```
Generación 2000
Secuencias ordenadas por calificacion
('TGGTTGCG-TTGCTTCTTCTGGCCATTT-CGCATCTGTA-TAAC', 595)
('TGGTTGCGTTGCTTCT-TCTGGCCA-TTTCGCATCTGTATAA-C', 518)
('TGGTTGCGTTG--CTTCTTCTGGCCATTTCGCA-TCTGTATAAC', 512)
('TGGTTGCGTTGCTTCTT-CTGGCCAT-TTCG-CATCTGTATAAC', 506)
('TGGTTGCGT-T-GCTTCTTCTGGCCATTTCGCATCTGT-ATAAC', 499)
('TGGTTGCGT-TGCTTCTTCTGGCCATTTCGCA-TCTGTAT-AAC', 478)
('TGG-TTGCGTT-GCTTCTTCTGGCCA-TTTCGCATCTGTATAAC', 205)
('-ATCAAAGATGAAATCTTCACCAAATGG-CGCATTTTCTT-CGATTTTATC', -5)
('ATCAAAGATGAAATCTT-CACCAAATGGCG-CATTTTCTTCGATTTTAT-C', -84)
('TTGATTCTTTAGCATCTGGTTTAAATCCAAC-TACCACGG-T-TTG', -102)
```

```
Padres Seleccionados
TGGTTGCG-TTGCTTCTTCTGGCCATTT-CGCATCTGTA-TAAC
TGGTTGCGTTGCTTCT-TCTGGCCA-TTTCGCATCTGTATAA-C

Hijos
Hijo 1:  TGGTTGCG-TTGCTTCT-TCTGGCCATTTCGCATCTGTA-TAA-C
Hijo 2:  TGGTTGCGTTGCTTCTTCTGGCCA-TTT-CGCATCTGTATAAC

Nueva poblacion
TGGTTGCG-TTGCTTCTTCTGGCCATTT-CGCATCTGTA-TAAC
TGGTTGCGTTGCTTCT-TCTGGCCA-TTTCGCATCTGTATAA-C
TGGTTGCGTTG--CTTCTTCTGGCCATTTCGCA-TCTGTATAAC
TGGTTGCGTTGCTTCTT-CTGGCCAT-TTCG-CATCTGTATAAC
TGGTTGCGT-T-GCTTCTTCTGGCCATTTCGCATCTGT-ATAAC
TGGTTGCGT-TGCTTCTTCTGGCCATTTCGCA-TCTGTAT-AAC
TGG-TTGCGTT-GCTTCTTCTGGCCA-TTTCGCATCTGTATAAC
-ATCAAAGATGAAATCTTCACCAAATGG-CGCATTTTCTT-CGATTTTATC
TGGTTGCG-TTGCTTCT-TCTGGC-CATTTCGC-AT-CTGTA-TAA-C
TGGTTGCGTTGCTTCTTCTGG-C-CA-TTT-CGCATC-TGTATAAC
```

Como podemos observar en la primera columna, hay un alineamiento (no completo) de las “T” y se logra apreciar en las siguientes columnas como se alinea, para esta ejecución se necesitaban mas generaciones para completar una buena alineación en las demás columnas.

Conclusión:

En este proyecto se realizan ejecuciones de manera aleatoria, pues es difícil si no imposible volver a obtener una misma población igual que la anterior ya que cada ejecución es distinta y con resultados diferentes.

Así podemos asegurarnos que puede funcionar con cualquier secuencia o mas secuencias de las elegidas en este proyecto.

El algoritmo genético en bioinformática representa una fusión fascinante entre la informática y la biología.

Esto no solo refleja la complejidad de los sistemas biológicos, sino que también demuestra la capacidad y la forma de utilizar la computación para modelar y resolver problemas de este ámbito.