



UNIVERSIDAD CARLOS III DE MADRID

# Inteligencia en Redes de Comunicaciones

Ingeniería de Telecomunicación

## Algoritmos Evolutivos y Algoritmos Genéticos



Trabajo realizado por:

Alfonso Mateos Andaluz

N.I.A.: 100027597

# ALGORITMOS EVOLUTIVOS Y ALGORITMOS GENÉTICOS

*“Así como la selección natural trabaja exclusivamente para y por el bien de cada ser viviente, todas las dotes mentales y corporales tienden a progresar en dirección a la perfección.”*

*Charles Darwin, El Origen de las Especies*

## **Introducción**

Muchos problemas de optimización que aparecen en los ámbitos de las ingenierías son muy difíciles de solucionar por medio de técnicas tradicionales, por lo que a menudo se aplican algoritmos evolutivos, inspirados en la naturaleza, que recogen un conjunto de modelos basados en la evolución de los seres vivos.

El inicio de la utilización de las estrategias evolutivas en la solución de este tipo de problemas data del año 1960 cuando John Holland planteó la posibilidad de incorporar los mecanismos naturales de selección y supervivencia a la resolución de problemas de Inteligencia Artificial (“Adaptation in Natural and Artificial Systems”). Esta investigación fue fundamentalmente académica, siendo su realización práctica en aquella época muy difícil. La simulación de procesos de evolución natural de las especies da como resultado una técnica de optimización estocástica que posteriormente fue llamada algoritmos evolutivos, y que fueron enmarcados dentro de las técnicas no convencionales de optimización para problemas del mundo real. A partir de la creación de estas estrategias evolutivas aparecieron otras vías de investigación como son: Algoritmos Genéticos (Goldberg), Programación Genética (Koza), Programación Evolutiva (Fogel), y Estrategias de Evolución (Rechenberg/Schwefel), en donde la estrategia de evolución más conocida hoy en día son los algoritmos genéticos.

Los algoritmos genéticos constituyen una técnica poderosa de búsqueda y optimización con un comportamiento altamente paralelo, inspirado en el principio darwiniano de selección natural y reproducción genética. En este principio de selección de los individuos más aptos, tienen mayor longevidad y por tanto mayor probabilidad de reproducción. Los individuos descendientes de estos individuos tienen una mayor posibilidad de transmitir sus códigos genéticos a las próximas generaciones.

La aparición de computadores de grandes prestaciones y bajo coste a mediados de los 80 permite aplicar los algoritmos evolutivos a la resolución de ciertos problemas de ingeniería que antes eran inabordables, y a partir de entonces el desarrollo de estas técnicas ha sido continuo. Actualmente los algoritmos genéticos han cobrado gran importancia por su potencial como una técnica importante para la solución de problemas complejos, siendo aplicados constantemente en la ingeniería. Las aplicaciones de los algoritmos genéticos han sido muy conocidas en áreas como diseño de circuitos, cálculo de estrategias de mercado, reconocimiento de patrones, acústica, ingeniería aeroespacial, astronomía y astrofísica, química, juegos, programación y secuenciación de operaciones, contabilidad lineal de procesos, programación de rutas, interpolación de superficies, tecnología de grupos, facilidad en diseño y localización, transporte de materiales y muchos otros problemas que involucran de alguna manera procesos de optimización.

## **Modelos de computación bio-inspirados**

En este apartado se pretende situar en un marco más general los temas que se tratarán en más profundidad, y el campo en el que nos situamos es el de la computación bio-inspirada.

La computación bio-inspirada se basa en emplear analogías con sistemas naturales o sociales para diseñar métodos heurísticos no determinísticos de “búsqueda”, de “aprendizaje”, de imitación de “comportamiento”, etc.

En la actualidad los algoritmos bio-inspirados (ABs) son uno de los campos más prometedores de investigación en el diseño de algoritmos.

Los algoritmos bioinspirados modelan (de forma aproximada) un fenómeno existente en la naturaleza, constituyendo así una metáfora biológica para resolver problemas. Son algoritmos no determinísticos, que a menudo presentan, implícitamente, una estructura paralela (múltiples agentes), y son adaptativos (utilizan realimentación con el entorno para modificar el modelo y los parámetros).

Algunos modelos de computación bioinspirados son:

- ✓ **Algoritmos evolutivos**
- ✓ Redes neuronales
- ✓ Algoritmos inmunológicos
- ✓ Algoritmos basados en inteligencia de enjambres (swarm intelligence) y dentro de ellos, los algoritmos basados en colonias de hormigas.

## **Computación evolutiva**

La computación evolutiva es una rama de la computación emergente que engloba técnicas que simulan la evolución natural, y constituye un enfoque alternativo para abordar problemas complejos de búsqueda y aprendizaje a través de modelos computacionales de procesos evolutivos.

Parte de un hecho observado en la naturaleza: los organismos vivos poseen una destreza consumada en la resolución de los problemas que se les presentan, y obtienen sus habilidades, casi sin proponérselo, a través del mecanismo de la evolución natural.

La evolución se produce, en casi todos los organismos, como consecuencia de dos procesos primarios: la selección natural y la reproducción (cruce). La evolución natural es un proceso de cambio sobre una población reproductiva que contiene variedades de individuos con algunas características heredables y en donde algunas variedades difieren en su aptitud (éxito reproductivo).



### **Fondo Histórico**

En 1859, Darwin publica su libro El origen de las especies que levantó agrias polémica en el mundo científico por las revolucionarias teorías que sostenían que las especies evolucionan acorde al medio, para adaptarse a éste.

De esta manera el universo pasaba de ser una creación de Dios estática y perfecta y se planteaba como un conjunto de individuos en constante competición y evolución para poder perpetuar su especie en el tiempo. Las especies se crean, evolucionan y desaparecen si no se adaptan de forma que solo los mejores, los más aptos, los que mejor se adapten al medio sobreviven para perpetuar sus aptitudes.

De acuerdo con esta visión de la evolución, la computación ve en dicho marco un claro proceso de optimización: se toman los individuos mejores adaptados –mejores soluciones temporales –, se cruzan –mezclan–, generando nuevos individuos –nuevas soluciones– que contendrán parte del código genético –información– de sus antecesores, y el promedio de adaptación de toda la población se mejora.

Originada a finales de los años 50 con los trabajos de Bremermann, Friedberg, Box y otros, el campo permaneció en desconocimiento por tres décadas debido a la ausencia de una plataforma computacional poderosa y defectos metodológicos de los primeros métodos (Fogel), hasta que la llegada de los nuevos trabajos de Holland, Rechenberg, Schwefel y Fogel cambiaron lentamente el escenario. Hoy en día el incremento de su empleo en la ciencia es exponencial.

La computación evolutiva se ha convertido en un concepto general adaptable para resolución de problemas, en especial problemas difíciles de optimización, gracias entre otros factores a que añade flexibilidad y adaptabilidad en la resolución, combinables con la robustez y las ventajas de la búsqueda global.



## **Computación Evolutiva y Optimización**

Un problema de optimización requiere hallar un conjunto de parámetros de forma que se cumpla un cierto criterio de calidad que se quiere optimizar, es decir, maximizando o minimizando una cierta función de evaluación  $f(x)$  dada.

Los algoritmos evolutivos son especialmente útiles cuando nos encontramos con problemas difíciles o altamente irresolubles, como lo son aquellos caracterizados por una alta dimensionalidad, multimodalidad, fuerte no linealidad, no diferenciabilidad, presencia de ruido y cuando se trata con funciones dependientes del tiempo.



## **Algoritmos Evolutivos**

Este término es empleado para describir sistemas de resolución de problemas de optimización o búsqueda basados en el ordenador empleando modelos computacionales de algún mecanismo de evolución conocido como elemento clave en su diseño e implementación.

Los algoritmos evolutivos trabajan con una población de individuos, que representan soluciones candidatas a un problema. Esta población se somete a ciertas transformaciones y después a un proceso de selección, que favorece a los mejores. Cada ciclo de transformación y selección constituye una generación, de forma que después de cierto número de generaciones se espera que el mejor individuo de la población esté cerca de la solución buscada. Los algoritmos evolutivos combinan la búsqueda aleatoria, dada por las transformaciones de la población, con una búsqueda dirigida dada por la selección.

Principales Componentes:

- ✓ Población de individuos, que son una representación (no necesariamente directa) de posibles soluciones.
- ✓ Procedimiento de selección basado en la aptitud de los individuos para resolver el problema.
- ✓ Procedimiento de transformación para construir nuevos individuos a partir de los anteriores.

La estructura general de los algoritmos evolutivos responde al siguiente esquema:

```

t :=0;
Inicializar P(t);
Evaluar P(t)
While no se termine do
begin
    P'(t) := variación [P(t)];
    evaluar [P'(t)];
    P(t+1) := seleccionar [P'(t) ∪ Q];
    t:=t+1;
end

```



## **Características**

La característica fundamental de los algoritmos evolutivos radica en los métodos de generación de soluciones: se parte de un conjunto de soluciones iniciales y se van empleando un conjunto de operadores de búsqueda para ir refinando la solución final. Para realizar dicho refinamiento de las soluciones, se pueden utilizar técnicas clásicas –como el seguimiento del gradiente (Hill Climbing) – complementadas con mecanismos biológicos de exploración: población de soluciones, operadores genéticos.

## **Clasificación**

Existen las siguientes modificaciones sobre el esquema general:

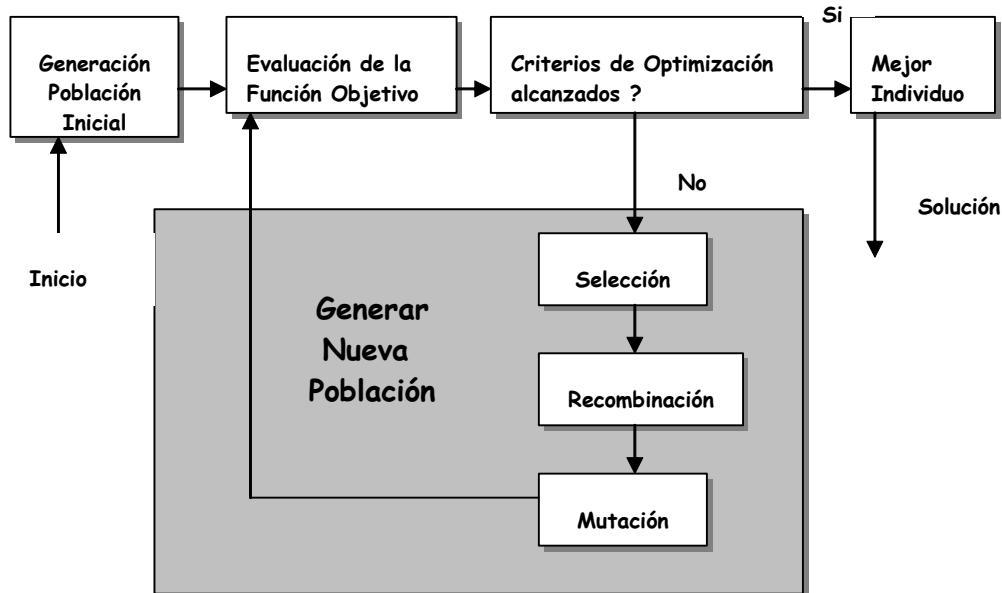
✓ **Estrategias Evolutivas:** Técnica desarrollada por Rechenberg y Schwefel y extendida por Herdy, Kursawe, Ostermeier, Rudolph, y otros, fue diseñada inicialmente con la meta de resolver problemas de optimización discretos y continuos, principalmente experimentales y considerados difíciles. Trabaja con vectores de números reales –con desviaciones estándar– que codifican las posibles soluciones de problemas numéricos. Utiliza recombinación o cruce (crossover aritmético), mutación y la operación de selección, ya sea determinística o probabilística, elimina las peores soluciones de la población y no genera copia de aquellos individuos con una aptitud por debajo de la aptitud promedio.

✓ **Programación Evolutiva:** Técnica introducida por Fogel y extendida por Burgin, Atmar y otros, inicialmente fue diseñada como un intento de crear inteligencia artificial. La representación del problema se realiza mediante números reales (cualquier estructura de datos), y emplea los mecanismos de mutación y selección. El procedimiento es muy similar a las estrategias evolutivas con la diferencia de que no emplea la recombinación, de tal forma que son denominadas en conjunto algoritmos evolutivos como una manera de diferenciarlas de los algoritmos genéticos.

✓ **Algoritmos Genéticos:** Modelan el proceso de evolución como una sucesión de frecuentes cambios en los genes, con soluciones análogas a cromosomas. Trabajan con una población de cadenas binarias para la representación del problema, y el espacio de soluciones posibles es explorado aplicando transformaciones a éstas soluciones candidatas tal y como se observa en los organismos vivientes: cruce, inversión y mutación. Como método de selección emplean en mecanismo de la ruleta (a veces con elitismo). Constituyen el paradigma más completo de la computación evolutiva ya que

resumen de modo natural todas las ideas fundamentales de dicho enfoque. Son muy flexibles ya que pueden adoptar con facilidad nuevas ideas, generales o específicas, que surjan dentro del campo de la computación evolutiva. Además, se pueden hibridar fácilmente con otros paradigmas y enfoques, aunque no tengan ninguna relación con la computación evolutiva. Se trata del paradigma con mayor base teórica.

NOTA: El diagrama de funcionamiento de los algoritmos genéticos responde al siguiente esquema:



✓ **Programación genética:** Desarrollada por Koza, los individuos son programas o autómatas de longitud variable, descritos mediante Expresiones-S de LISP representadas habitualmente como árboles, y como operadores de variación emplea crossover y modificación, además de mecanismos de selección.

Como resumen de las técnicas comentadas se presenta la siguiente tabla resumen:

Algoritmo	Representación del problema	Operadores de variación	Métodos de selección
<b>Algoritmos genéticos</b> (Goldberg)	Cadena binaria	Mutación y crossover	Selección de rueda de ruleta (a veces con elitismo)
<b>Estrategias de Evolución</b> (Rechenberg/Schwefel)	Vector de reales + desviaciones estándar	Mutación gaussiana y crossover aritmético (diferentes tipos)	Diferentes tipos de selección: lambda,mu; lambda+mu...
<b>Programación evolutiva</b> (Fogel)	Números reales	Mutación	Diversos tipos de selección
<b>Programación genética</b> (Koza)	Expresiones-S de LISP representadas habitualmente como árboles	Crossover, algo de mutación	Diversos tipos de selección

## **Algoritmos genéticos**



### **Estructura general**

Los algoritmos genéticos son estrategias de búsqueda estocástica basados en el mecanismo de selección natural y en algunos casos se involucran aspectos de genética natural, imitando a la evolución biológica como estrategia para resolver problemas. Los algoritmos genéticos difieren de las estrategias de búsqueda convencionales en que estos (los AGs) trabajan sobre un conjunto de potenciales soluciones, llamado *población*. Esta población está compuesta de una serie de soluciones llamadas *individuos* y un individuo está conformado por una serie de posiciones que representan cada una de las variables involucradas en los procesos de optimización y que son llamados *cromosomas*. Estos cromosomas están compuestos por una cadena de símbolos que en muchos casos está presentada en números binarios. En un algoritmo genético cada individuo está definido como una estructura de datos que representa una posible solución del espacio de búsqueda del problema. Las estrategias de evolución trabajan sobre los individuos, que representan las soluciones del problema, por lo que estos *evolucionan* a través de *generaciones*. Dentro de la población cada individuo es diferenciado de acuerdo con su valor de *aptitud*, que es obtenido usando algunas medidas de acuerdo con el problema a resolver. Para la obtención de las próximas generaciones se crean nuevos individuos, llamados *hijos*, utilizando dos estrategias de evolución básicas como son el operador de cruce y el de mutación (empleadas generalmente de forma aleatoria).

Una nueva generación es obtenida mediante la utilización del operador de selección, que básicamente se realiza sobre los valores de aptitud de los individuos, sobre la población obtenida tras el cruce y la mutación. El operador de selección debe mantener constante el número de individuos de la población, y a través de las generaciones el individuo con mayor valor de aptitud tiene mayores posibilidades de ser seleccionado en la siguiente generación. Entre el conjunto de soluciones candidatas generadas aleatoriamente muchas no funcionarán en absoluto y serán eliminadas; sin embargo, por puro azar, unas pocas pueden resultar prometedoras y pueden mostrar actividad, aunque sólo sea actividad débil e imperfecta, hacia la solución del problema. Estas candidatas prometedoras se conservan y se les permite reproducirse. Se realizan múltiples copias de ellas, pero las copias no son perfectas; se introducen cambios aleatorios durante el proceso de copia. Esta descendencia prosigue con la siguiente generación, formando un nuevo acervo de soluciones candidatas que son nuevamente sometidas a una ronda de evaluación de aptitud. Las candidatas que han empeorado o no han mejorado con los cambios en su código son eliminadas de nuevo; pero, de nuevo por puro azar, las variaciones aleatorias introducidas en la población pueden haber mejorado a algunos individuos, convirtiéndolos en mejores soluciones del problema, más completas o más eficientes. De nuevo, se seleccionan y copian estos individuos vencedores hacia la siguiente generación con cambios aleatorios, y el proceso se repite. Las expectativas son que la aptitud media de la población se incrementará en cada ronda y, por tanto, repitiendo este proceso cientos o miles de rondas, pueden descubrirse soluciones muy buenas del problema; en otras palabras, *tras varias iteraciones el algoritmo converge al individuo con mejor valor de aptitud, el cual representará el óptimo o subóptimo del problema.*



## **Vocabulario en Algoritmos Genéticos**

Los algoritmos genéticos toman para su definición terminología utilizada en Genética Natural y Ciencia Computacional, por lo que la literatura específica es una combinación entre los términos naturales y los artificiales, por esto se debe tener en cuenta:

- La Estructura de codificación con la cual se construyen las soluciones para el problema se llaman *cromosomas*. Uno o más cromosomas pueden ser requeridos para conformar una población.
- El conjunto completo de cromosomas se llama *genotipo* y un individuo en particular, un organismo resultante, se llama *fenotipo*.
- Cada cromosoma contiene una serie de valores individuales llamados *genes*.
- Cada gen contiene información valiosa de una variable en particular y su ubicación dentro del individuo se conoce como *loco*.
- Los diferentes valores de un gen son llamados *alelos*.



## **Componentes de un Algoritmo Genético**

Los algoritmos genéticos se pueden caracterizar a través de los siguientes componentes:



### **Definición del *problema* a optimizar:**

Los algoritmos genéticos tienen su campo de aplicación importante en problemas de optimización complejos, donde se tiene diferentes parámetros o conjuntos de variables que deben ser combinadas para su solución. También se enmarcan en este campo problemas con muchas restricciones y problemas con espacios de búsqueda muy grandes. Este tipo de optimización difiere en muchos aspectos de los procesos convencionales de optimización, ya que los algoritmos genéticos:

- Trabajan sobre un conjunto codificado de soluciones, no sobre las soluciones mismas.
- Trabajan sobre un conjunto de soluciones, no sobre una solución única.
- Utilizan información simple para determinar la aptitud de los individuos, dejando de lado los procesos de derivación u otra información adicional.
- Usan procesos de transición probabilística para la búsqueda de la solución, no reglas determinísticas.

Gracias a estas características, los algoritmos genéticos no requieren información matemática adicional sobre optimización, por lo que pueden tomar otro tipo de funciones objetivo y todo tipo de restricciones (lineales y no lineales) definidas sobre espacios discretos, continuos o espacios de búsqueda combinados.

La robustez de los operadores de evolución hace muy efectivos a los algoritmos genéticos en procesos de búsqueda de soluciones globales. A diferencia de las soluciones obtenidas con un algoritmo genético, las búsquedas tradicionales operan sobre una solución particular del problema, buscando soluciones óptimas en las cercanías de esta solución particular, en muchos casos no encontrando soluciones globales para el problema. Por ello, los algoritmos genéticos proveen una gran flexibilidad para ser combinados con heurísticas específicas para la solución de un problema en particular.



### **Representación para la solución del problema:**

Para la solución de un problema en particular es importante definir una estructura del cromosoma de acuerdo con el espacio de búsqueda. En el caso de los algoritmos genéticos la representación más utilizada es la representación binaria, debido a su facilidad de manipulación por los operadores genéticos, su fácil transformación en números enteros o reales, además de la facilidad que da para la demostración de teoremas.

La codificación de números reales en cadenas de números binarios se debe hacer teniendo en cuenta el siguiente proceso y la precisión requerida para la solución del problema:

$x_i \in [a_i, b_i]$  en donde el rango de la variable debe quedar dividida en  $(b_i - a_i) \cdot 10^p$ , donde  $p$  es la precisión requerida por el proceso (determinando así su potencia)

La longitud de la cadena binaria de caracteres denotada por  $m_i$ , para cada variable es calculada como  $2^{m_i-1} < (b_i - a_i) \cdot 10^p < 2^{m_i} - 1$

### **Decodificación del cromosoma:**

Para determinar el número real que la cadena binaria de caracteres representa. Es importante tener en cuenta que la cadena no representa un número real, sino que este número binario etiqueta un número dentro del intervalo inicialmente fijado.

### **Evaluación de un individuo:**

Nos muestra el valor de aptitud de cada uno de los individuos. Esta aptitud viene dada por una función que es la unión entre el mundo natural y el problema a resolver matemáticamente. Esta función de aptitud es particular para cada problema particular a resolver y representa para un algoritmo genético lo que el medio ambiente representa para los humanos.

## **Operadores Evolutivos: Selección**

Un algoritmo genético puede utilizar muchas técnicas diferentes para seleccionar a los individuos que deben copiarse hacia la siguiente generación, pero abajo se listan algunos de los más comunes. Algunos de estos métodos son mutuamente exclusivos, pero otros pueden utilizarse en combinación, algo que se hace a menudo.

Las diferentes variantes en la operación de selección son:

- ✓ **Selección elitista:** se garantiza la selección de los miembros más aptos de cada generación. (La mayoría de los AGs no utilizan elitismo puro, sino que usan una forma modificada por la que el individuo mejor, o algunos de los mejores, son copiados hacia la siguiente generación en caso de que no surja nada mejor).
- ✓ **Selección proporcional a la aptitud:** los individuos más aptos tienen más probabilidad de ser seleccionados, pero no la certeza.
- ✓ **Selección por rueda de ruleta:** una forma de selección proporcional a la aptitud en la que la probabilidad de que un individuo sea seleccionado es proporcional a la diferencia entre su aptitud y la de sus competidores. (Conceptualmente, esto puede representarse como un juego de ruleta: cada individuo obtiene una sección de la ruleta, pero los más aptos obtienen secciones mayores que las de los menos aptos. Luego la ruleta se hace girar, y en cada vez se elige al individuo que “posea” la sección en la que se pare la ruleta).

- ✓ **Selección escalada:** al incrementarse la aptitud media de la población, la fuerza de la presión selectiva también aumenta y la función de aptitud se hace más discriminadora. Este método puede ser útil para seleccionar más tarde, cuando todos los individuos tengan una aptitud relativamente alta y sólo les distingan pequeñas diferencias en la aptitud.
- ✓ **Selección por torneo:** se eligen subgrupos de individuos de la población, y los miembros de cada subgrupo compiten entre ellos. Sólo se elige a un individuo de cada subgrupo para la reproducción.
- ✓ **Selección por rango:** a cada individuo de la población se le asigna un rango numérico basado en su aptitud, y la selección se basa en este ranking, en lugar de las diferencias absolutas en aptitud. La ventaja de este método es que puede evitar que individuos muy aptos ganen dominancia al principio a expensas de los menos aptos, lo que reduciría la diversidad genética de la población y podría obstaculizar la búsqueda de una solución aceptable.
- ✓ **Selección generacional:** la descendencia de los individuos seleccionados en cada generación se convierte en toda la siguiente generación. No se conservan individuos entre las generaciones.
- ✓ **Selección por estado estacionario:** la descendencia de los individuos seleccionados en cada generación vuelven al acervo genético preexistente, reemplazando a algunos de los miembros menos aptos de la siguiente generación. Se conservan algunos individuos entre generaciones.
- ✓ **Selección jerárquica:** los individuos atraviesan múltiples rondas de selección en cada generación. Las evaluaciones de los primeros niveles son más rápidas y menos discriminatorias, mientras que los que sobreviven hasta niveles más altos son evaluados más rigurosamente. La ventaja de este método es que reduce el tiempo total de cálculo al utilizar una evaluación más rápida y menos selectiva para eliminar a la mayoría de los individuos que se muestran poco o nada prometedores, y sometiendo a una evaluación de aptitud más rigurosa y computacionalmente más costosa sólo a los que sobreviven a esta prueba inicial.





## **Operadores Genéticos: Cruce y Mutación**

Una vez que la selección ha elegido a los individuos aptos, éstos deben ser alterados aleatoriamente con la esperanza de mejorar su aptitud para la siguiente generación. Existen dos estrategias básicas para llevar esto a cabo. La primera y más sencilla se llama *mutación*. Al igual que una mutación en los seres vivos cambia un gen por otro, una mutación en un algoritmo genético también causa pequeñas alteraciones en puntos concretos del código de un individuo.

El segundo método se llama *cruce*, e implica elegir a dos individuos para que intercambien segmentos de su código, produciendo una “descendencia” artificial cuyos individuos son combinaciones de sus padres. Este proceso pretende simular el proceso análogo de la recombinación que se da en los cromosomas durante la reproducción sexual. Las formas comunes de cruzamiento incluyen al cruzamiento de un punto, en el que se establece un punto de intercambio en un lugar aleatorio del genoma de los dos individuos, y uno de los individuos contribuye todo su código anterior a ese punto y el otro individuo contribuye todo su código a partir de ese punto para producir una descendencia, y al cruzamiento uniforme, en el que el valor de una posición dada en el genoma de la descendencia corresponde al valor en esa posición del genoma de uno de

los padres o al valor en esa posición del genoma del otro padre, elegido con un 50% de probabilidad.

Los operadores genéticos en este caso son operadores con memoria, guardando en cada iteración los códigos genéticos de los mejores individuos a través de las generaciones. Los procesos de evolución darviniana generalmente son procesos miméticos a través de las generaciones.

Cruce y mutación	
Efecto de estos dos operadores genéticos en los individuos de una población de cadenas de 8 bits.	
	Proceso de cruce: el punto de intercambio se establece entre las posiciones quinta y sexta del genoma, produciendo un nuevo individuo que es híbrido de sus progenitores.
	Individuo sufriendo una mutación en la posición 4, cambiando el 0 de esa posición de su genoma por un 1.

## **Ventajas de los Algoritmos genéticos**

- El primer y más importante punto es que los algoritmos genéticos son intrínsecamente *paralelos*. La mayoría de los otros algoritmos son en serie y sólo pueden explorar el espacio de soluciones hacia una solución en una dirección al mismo tiempo, y si la solución que descubren resulta subóptima, no se puede hacer otra cosa que abandonar todo el trabajo hecho y empezar de nuevo. Sin embargo, ya que los AGs tienen descendencia múltiple, pueden explorar el espacio de soluciones en múltiples direcciones a la vez. Si un camino resulta ser un callejón sin salida, pueden eliminarlo fácilmente y continuar el trabajo en avenidas más prometedoras, dándoles una mayor probabilidad en cada ejecución de encontrar la solución. Otra ventaja del paralelismo es que, al evaluar la aptitud de una cadena particular, un algoritmo genético sondea al mismo tiempo cada uno de los espacios a los que pertenece dicha cadena. Tras muchas evaluaciones, iría obteniendo un valor cada vez más preciso de la aptitud media de cada uno de estos espacios, cada uno de los cuales contiene muchos miembros. Por tanto, un AG que evalúe explícitamente un número pequeño de individuos está evaluando implícitamente un grupo de individuos mucho más grande. De la misma manera, el AG puede dirigirse hacia el espacio con los individuos más aptos y encontrar el mejor de ese grupo. En el contexto de los algoritmos evolutivos, esto se conoce como teorema del esquema, y es la ventaja principal de los AGs sobre otros métodos de resolución de problemas.
- Debido al paralelismo que les permite evaluar implícitamente muchos esquemas a la vez, los algoritmos genéticos funcionan particularmente bien resolviendo problemas cuyo espacio de soluciones potenciales es realmente grande, demasiado vasto para hacer una búsqueda exhaustiva en un tiempo razonable. La mayoría de los problemas que caen en esta categoría se conocen como “no lineales”. En un problema lineal, la aptitud de cada componente es independiente, por lo que cualquier mejora en alguna parte dará como resultado una mejora en el sistema completo. No es necesario decir que hay pocos

problemas como éste en la vida real. La no linealidad es la norma, donde cambiar un componente puede tener efectos en cadena en todo el sistema, y donde cambios múltiples que, individualmente, son perjudiciales, en combinación pueden conducir hacia mejoras en la aptitud mucho mayores. La no linealidad produce una explosión combinatoria en el número de posibilidades. El paralelismo implícito de los AGs les permite superar incluso este enorme número de posibilidades, y encontrar con éxito resultados óptimos o muy buenos en un corto periodo de tiempo, tras muestrear directamente sólo regiones pequeñas del vasto paisaje adaptativo.

- Otra ventaja notable de los algoritmos genéticos es que se desenvuelven bien en problemas con un paisaje adaptativo complejo: aquéllos en los que la función de aptitud es discontinua, ruidosa, cambia con el tiempo, o tiene muchos óptimos locales. La mayoría de los problemas prácticos tienen un espacio de soluciones enorme, imposible de explorar exhaustivamente; el reto se convierte entonces en cómo evitar los óptimos locales, es decir, las soluciones que son mejores que todas las que son similares a ella, pero que no son mejores que otras soluciones distintas situadas en algún otro lugar del espacio de soluciones. Muchos algoritmos de búsqueda pueden quedar atrapados en los óptimos locales. Los algoritmos evolutivos, por otro lado, han demostrado su efectividad al escapar de los óptimos locales y descubrir el óptimo global incluso en paisajes adaptativos muy escabrosos y complejos. Los cuatro componentes principales de los AGs -paralelismo, selección, mutación y cruzamiento- trabajan juntos para conseguir esto.

- Otra área en el que destacan los algoritmos genéticos es su habilidad para manipular muchos parámetros simultáneamente. Muchos problemas de la vida real no pueden definirse en términos de un único valor que hay que minimizar o maximizar, sino que deben expresarse en términos de múltiples objetivos, a menudo involucrando contrapartidas: uno sólo puede mejorar a expensas de otro. Los AGs son muy buenos resolviendo estos problemas: en particular, su uso del paralelismo les permite producir múltiples soluciones, igualmente buenas, al mismo problema, donde posiblemente una solución candidata optimiza un parámetro y otra candidata optimiza uno distinto, y luego un supervisor humano puede seleccionar una de esas candidatas para su utilización. Si una solución particular a un problema con múltiples objetivos optimiza un parámetro hasta el punto en el que ese parámetro no puede mejorarse más sin causar una correspondiente pérdida de calidad en algún otro parámetro, esa solución se llama óptimo paretiano o no dominada.

- Finalmente, una de las cualidades de los algoritmos genéticos que, a primera vista, puede parecer un desastre, resulta ser una de sus ventajas: a saber, los AGs no saben nada de los problemas que deben resolver. En lugar de utilizar información específica conocida a priori para guiar cada paso y realizar cambios guiados expresamente hacia la mejora, los AGs realizan cambios aleatorios en sus soluciones candidatas y luego utilizan la función de aptitud para determinar si esos cambios producen una mejora. Como sus decisiones están basadas en la aleatoriedad, todos los caminos de búsqueda posibles están abiertos teóricamente a un AG; en contraste, cualquier estrategia de resolución de problemas que dependa de un conocimiento previo, debe inevitablemente comenzar descartando muchos caminos a priori, perdiendo así cualquier solución novedosa que pueda existir. Los AGs, al carecer de ideas preconcebidas basadas en creencias establecidas sobre “cómo deben hacerse las cosas” o sobre lo que “de ninguna manera podría funcionar”, los AGs no tienen este problema. De manera similar, cualquier técnica que dependa de conocimiento previo fracasará cuando no esté disponible tal conocimiento, pero, de nuevo, los AGs no se ven afectados negativamente por la ignorancia. Mediante sus componentes de paralelismo, cruzamiento y mutación,

pueden viajar extensamente por el paisaje adaptativo, explorando regiones que algoritmos producidos con inteligencia podrían no haber tenido en cuenta, y revelando potencialmente soluciones asombrosas y creativas.



## **Limitaciones de los Algoritmos Genéticos**

Aunque los algoritmos genéticos han demostrado su eficiencia y potencia como estrategia de resolución de problemas, no son la panacea. Los AGs tienen ciertas limitaciones; sin embargo, se demostrará que todas ellas pueden superarse y que ninguna de ellas afecta a la validez de la evolución biológica.

- La primera y más importante consideración al crear un algoritmo genético es definir una representación del problema. El lenguaje utilizado para especificar soluciones candidatas debe ser robusto; es decir, debe ser capaz de tolerar cambios aleatorios que no produzcan constantemente errores fatales o resultados sin sentido. Hay dos maneras principales para conseguir esto. La primera, utilizada por la mayoría de los algoritmos genéticos, es definir a los individuos como listas de números -binarios, enteros o reales- donde cada número representa algún aspecto de la solución candidata. En otro método, la programación genética, el propio código del programa sí cambia. Ambos métodos producen representaciones robustas ante la mutación, y pueden representar muchos tipos diferentes de problema con éxito.
- El problema de cómo escribir la función de aptitud debe considerarse cuidadosamente para que se pueda alcanzar una mayor aptitud y verdaderamente signifique una solución mejor para el problema dado. Si se elige mal una función de aptitud o se define de manera inexacta, puede que el algoritmo genético sea incapaz de encontrar una solución al problema, o puede acabar resolviendo el problema equivocado (esta última situación se describe a veces como la tendencia del AG a “engañar”, aunque en realidad lo que está pasando es que el AG está haciendo lo que se le pidió hacer, no lo que sus creadores pretendían que hiciera).
- Además de elegir bien la función de aptitud, también deben elegirse cuidadosamente los otros parámetros de un AG, como son el tamaño de la población, el ritmo de mutación y cruzamiento, el tipo y fuerza de la selección. Si el tamaño de la población es demasiado pequeño, puede que el algoritmo genético no explore suficientemente el espacio de soluciones para encontrar buenas soluciones consistentemente. Si el ritmo de cambio genético es demasiado alto o el sistema de selección se escoge inadecuadamente, puede alterarse el desarrollo de esquemas beneficiosos y la población puede entrar en catástrofe de errores, al cambiar demasiado rápido para que la selección llegue a producir convergencia. La solución ha sido “la evolución de la evolutividad”: las adaptaciones que alteran la habilidad de una especie para adaptarse.
- Un problema con el que los algoritmos genéticos tienen dificultades son los problemas con las funciones de aptitud “engañosas”, en las que la situación de los puntos mejorados ofrecen información engañosa sobre dónde se encuentra probablemente el óptimo global. La solución a este problema es la misma para los algoritmos genéticos y la evolución biológica: la evolución no es un proceso que deba encontrar siempre el óptimo global. Puede funcionar casi igual de bien alcanzando la una solución muy buena aunque no sea la realmente óptima y, para la mayoría de las situaciones, eso será suficiente, incluso aunque el óptimo global no pueda alcanzarse fácilmente desde ese punto.
- Un problema muy conocido que puede surgir con un AG se conoce como convergencia prematura. Si un individuo que es más apto que la mayoría de sus

competidores emerge muy pronto en el curso de la ejecución, se puede reproducir tan abundantemente que merme la diversidad de la población demasiado pronto, provocando que el algoritmo converja hacia el óptimo local que representa ese individuo, en lugar de rastrear el paisaje adaptativo lo bastante a fondo para encontrar el óptimo global. Esto es un problema especialmente común en las poblaciones pequeñas, donde incluso una variación aleatoria en el ritmo de reproducción puede provocar que un genotipo se haga dominante sobre los otros. Los métodos más comunes implementados por los investigadores en AGs para solucionar este problema implican controlar la fuerza selectiva, para no proporcionar tanta ventaja a los individuos excesivamente aptos. La selección escalada, por rango y por torneo son tres de los métodos principales para conseguir esto.

- Finalmente, varios investigadores aconsejan no utilizar algoritmos genéticos en problemas resolubles de manera analítica. No es que los algoritmos genéticos no puedan encontrar soluciones buenas para estos problemas; simplemente es que los métodos analíticos tradicionales consumen mucho menos tiempo y potencia computacional que los AGs y, a diferencia de los AGs, a menudo está demostrado matemáticamente que ofrecen la única solución exacta.