NÚMEROS
Revista de Didáctica de las Matemáticas
http://www.sinewton.org/numeros
ISSN: 1887-1984
Volumen 71, agosto de 2009, páginas 29–47

Algoritmos Genéticos. Una visión práctica

Belén Melián Batista (Universidad de La Laguna)
José A. Moreno Pérez (Universidad de La Laguna)
J. Marcos Moreno Vega (Universidad de La Laguna)

Fecha de recepción: 7 de septiembre de 2009
Artículo solicitado al autor por la revista

Resumen Los algoritmos genéticos son métodos de optimización inspirados en la teoría de la evolución natural originada a partir de los estudios de Darwin. Existe un soporte matemático importante sobre su comportamiento y ha conseguido aplicaciones de éxito en la mayoría de los campos de aplicación. En este trabajo describimos los fundamentos de los algoritmos genéticos, las características básicas de los modelos más simples y los elementos que definen las versiones más relevantes.

Palabras clave Algoritmos Genéticos. Optimización. Metaheurísticas. Evolución. Aplicaciones.

Abstract Genetic algorithms are optimization methods inspired by the natural evolution theory originated from the studies of Darwin. There is an important mathematical support on their performance and they have achieved successful applications in most application fields. In this paper we describe the basics of genetic algorithms, the basic characteristics of the simplest models and the elements that define the relevant versions.

Keywords Genetic Algorithms. Optimization. Metaheuristics. Evolution. Aplication.

1. Introducción

Los *Algoritmos Genéticos* (AG's) son métodos de optimización basados en una simulación parcial de los mecanismos de la evolución natural. Están basados en la teoría de la evolución que surge con las investigaciones de Charles Darwin (Darwin, 1859). Los algoritmos genéticos fueron creados en la década de los 60's por John Holland (Holland, 1975), como un modelo para el estudio del fenómeno de adaptación natural y para el desarrollo de mecanismos que permitieran incorporar este fenómeno a los sistemas de cómputo. Los algoritmos genéticos alcanzaron popularidad a raíz de la publicación del libro de Goldberg (1989) que ponía las bases fuertes para su aplicación en problemas prácticos. Los algoritmos evolutivos constituyen una parte importante de la Computación Evolutiva, un área de la Inteligencia Artificial en constante crecimiento. Actualmente son innumerables las aplicaciones exitosas en las más diversas áreas industriales, comerciales y de la ingeniería (Goldberg y Sastry, 2008).

1.1. Un poco de biología

La *evolución biológica* es el proceso de transformación continua de las especies a través de cambios producidos en sucesivas generaciones. Esta evolución es la que da lugar a la aparición de nuevas especies pero también la que permite su adaptación a distintos ambientes con cambios en las características genéticas de los individuos. La evolución biológica se produce básicamente por dos procesos: la *selección de individuos* de la población según sus características y la *alteración genética* de los cromosomas que almacenan las características de la especie. La selección natural puede ser de dos categorías: la *selección reproductiva* hace que los individuos de ciertas características tengan mayor probabilidad de intervenir en los procesos de reproducción y la *selección ecológica* que hace que los individuos con ciertas características tengan mayor probabilidad de supervivencia. En ambos casos, las características que dan mayores probabilidades son las que favorecen la adaptación al entorno: mayor capacidad para obtener y procesar el alimento, escapar de los depredadores y otros peligros, resistir a las fluctuaciones ambientales, etc. La alteración genética de los individuos de una población tiene lugar durante la *reproducción* de los individuos, o cuando éstos sufren algún tipo de *mutación*. En la reproducción, los individuos intercambian material cromosómico; y en la mutación, se altera parte de la información de los cromosomas. La recombinación genética es el proceso mediante el cual la información genética se redistribuye entre dos cromosomas durante los procesos reproductivos y se transmiten a la descendencia. La mutación es un cambio permanente y transmisible en material genético de un individuo que se producen esporádicamente, muchas veces ligado a los procesos de reproducción.

En la naturaleza, estos procesos ocurren sobre una generación, y luego sobre su descendencia, y a continuación sobre la descendencia de ésta, y así sucesivamente. Después de cada ciclo, la generación actual será mejor que las anteriores, en el sentido de que los individuos estarán más evolucionados y más adaptados al medio.

2.1. Un poco de historia

Las tres figuras históricas claves en la aparición de los algoritmos genéticos, aparte de su creador John Hollad son, Darwin, Mendel y De Vries.

Charles Robert Darwin. El 27 de diciembre de 1831 zarpa de Davenport (Inglaterra) el Beagle, buque al mando del capitán Fitz-Roy, con varios objetivos científicos: completar el estudio de las costas de la Patagonia y Tierra del Fuego, realizar la cartografía de las costas de Chile, Perú y algunas islas del Pacífico y realizar varias observaciones cronométricas alrededor del mundo. A bordo viaja el joven naturalista de 22 años, Charles Robert Darwin (1809-1882). El 6 de enero de 1832 el Beagle fondea frente a las costas de Tenerife. Sin embargo, las autoridades obligan a la tripulación a permanecer a bordo del buque por miedo a que padezca el cólera. La larga travesía del Beagle, que atraca el 2 de octubre en Falmouth, sirve a Darwin para enunciar su teoría acerca del origen y evolución de las especies. Uno de los elementos claves en la teoría de Darwin es que la evolución se debe a la selección natural. Es decir, aquellos individuos más adaptados al medio tienen mayor probabilidad de sobrevivir y, de esta forma, las características que les hacen mejores se propagan entre la descendencia. De esta forma, y de manera gradual, surgen las diversas especies.

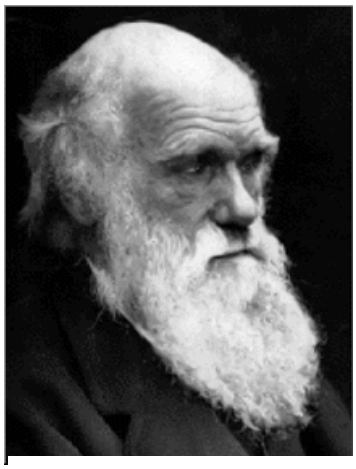


Figura 1. Charles Darwin

Gregor Johann Mendel: El monje agustino austriaco Johann Mendel (1822-1884) se interesó por los principios que rigen la herencia de características en las especies. En 1843 se ordenó sacerdote y diez años más tarde fue nombrado profesor suplente de la escuela moderna de Brno, lugar donde pasó la mayor parte de su vida. En el jardín del mencionado convento cultivó algunas variedades de guisantes. Escogió algunas características con alternativas claras. Por ejemplo, semillas redondas o rugosas. Seleccionó variedades de guisantes que producían descendencias homogéneas para estas características y estudió sus sucesivas descendencias. De esta forma, pudo enunciar sus leyes acerca de la herencia. Estas muestran en qué proporción se manifiestan las alternativas de cada característica. Para explicar las proporciones observadas, Mendel enunció la hipótesis de que la primera generación de guisantes contenía elementos hereditarios para ambas alternativas del carácter. Sus trabajos pueden considerarse como la base de la genética.

Figura 2. Gregor Mendel

Hugo De Vries: El botánico holandés Hugo De Vries (1848-1935) tuvo un papel importante en la difusión de los estudios de Darwin y de Mendel. En 1889 recuperó explicaciones propuestas por Darwin relacionados con la herencia genética. Por otro lado, en 1900, dieciséis años después de la muerte de Mendel, encontró sus trabajos y dio a conocer sus resultados, aunque en un principio no mencionó a Mendel. La contribución más importante de De Vries fue la introducción del concepto de mutación en la explicación de la evolución de las especies. Defendió que las especies no surgían de manera gradual por procesos de selección natural, sino a través de mutaciones de especies conocidas. Si estas mutaciones derivan en características beneficiosas, las mismas se propagan entre la descendencia. Sin embargo, sus teorías fueron abandonadas cuando se probó matemáticamente que las leyes de Mendel de la herencia genética y la deriva genética permitían explicar la evolución de las especies.

Figura 3. Hugo De Vries

John Holland: A John Holland (1929-) se le reconoce la paternidad de los Algoritmos Genéticos. Actualmente es profesor de Inteligencia Artificial de los Departamento de Psicología y miembro activo del Laboratorio de Inteligencia Artificial del Departamento Ingeniería Eléctrica y Ciencias de la Computación de la Universidad de Míchigan. En 1975 defendió su tesis doctoral "*Adaptation in Natural and Artificial Systems*" (Adaptación en Sistemas Naturales y Artificiales) en la Universidad de Michigan, la primera tesis doctoral en Ciencias de la Computación en dicha universidad. En ella, se proponía por primera vez una clase de métodos, llamados Algoritmos Genéticos, para la resolución de problemas. La idea que subyace en los algoritmos genéticos es que es posible implementar, en un ordenador, un programa que, guiado por los principios de la herencia y la evolución de las especies, suministre la solución de un problema.

Figura 4. John Holland

Los algoritmos constituyen una de las áreas más prometedoras de La Inteligencia Artificial y se incluye en amplia relación de técnicas y métodos de las Ciencias de la Computación inspirados en la biología. En particular, otros métodos de la resolución inteligente de problemas de optimización inspirados en la naturaleza que han alcanzado éxitos notables y un cierto prestigio en el área son: Las Redes Neuronales, Los Sistemas basados en Colonia de Hormigas, El Recocido Simulado, la Optimización por Enjambre, La Computación de Membranas y la Optimización Extrema.

2. Problemas de Optimización

Un problema de optimización consiste, desde el punto de vista matemático, en encontrar dónde se alcanza el óptimo de una función real. Por óptimo se entiende máximo o mínimo, según el caso, y se puede transformar uno en el otro por un cambio de signo en la función. Este óptimo no tiene por que ser único y el interés puede estar en encontrar todos los valores de la función donde se alcance el óptimo o sólo uno de ellos; incluso puede ser suficiente con acercarse a la optimalidad en un alto grado. Formalmente, dada la función $f: X \rightarrow \Re$ a minimizar, se trata de encontrar el $x^* \in X$ tal que $f(x^*) \leq f(x)$, $\forall\, x \in X$. Se dice que x^* es la solución óptima del problema:

$$\min f(x): x \in X.$$

En el caso más general los elementos $x \in X$ tienen varias componentes y el conjunto X es equivalente a un subconjunto de un espacio numérico multidimensional con el que se identifica. En este caso, el conjunto X se describe como el de los puntos de cierto espacio $\Re^n$ que verifican ciertas condiciones que se expresan mediante desigualdades e igualdades que, en general, se representan conjuntamente por la condición $g(x) \leq 0$ donde g y 0 son multidimensionales. Por tanto el problema se representa formalmente por:

$$\min f(x): g(x) \leq 0.$$

Y haciendo explícitas las componentes de x y de g el problema es:

$$\min f(x_1, x_2, \ldots, x_n): g_i(x_1, x_2, \ldots, x_n) \leq 0,\ i = 1, 2, \ldots, m.$$

Finalmente, la función objetivo f también puede ser vectorial lo que da lugar al problema más general:

$$\min f_1(x_1, x_2, \ldots, x_n), f_2(x_1, x_2, \ldots, x_n), \ldots, f_k(x_1, x_2, \ldots, x_n)$$

$$\text{sujeto a: } g_i(x_1, x_2, \ldots, x_n) \leq 0,\ i = 1, 2, \ldots, m.$$

El término *Optimización Global* corresponde al estudio de aquellos problemas donde las variables pueden variar continuamente en un subconjunto de la recta real y se denominan variables continuas. Además, en optimización global el papel de las restricciones se suele reducir a acotaciones en su campo de variación siendo la complejidad de la función objetivo la que juega el papel fundamental. Sin embargo, en los problemas que surgen en aplicaciones prácticas en campos de la ingeniería, la gestión, la técnica y la ciencia las variables sólo pueden tomar valores en un conjunto finito (o a lo sumo numerable). En estos problemas, las restricciones juegan un papel fundamental que hace que las soluciones factibles representen unas combinaciones que pueden ser muy difíciles de conseguir, casi tanto como optimizar la función objetivo, y constituyen el área de la *Optimización Combinatoria*. En éste área las variables suelen tomar valores enteros, y en muchos casos sólo los valores 0 y 1.

3. Optimización con algoritmo genético

Un algoritmo genético, desde el punto de vista de la optimización, es un método poblacional de búsqueda dirigida basada en probabilidad. Bajo una condición muy débil (que el algoritmo mantenga *elitismo*, es decir, guarde siempre al mejor elemento de la población sin hacerle ningún cambio) se puede demostrar que el algoritmo converge en probabilidad al óptimo. En otras palabras, al aumentar el número de iteraciones, la probabilidad de tener el óptimo en la población tiende a 1 (uno). Un algoritmo genético emula el comportamiento de una población de individuos que representan soluciones y que evoluciona en base a los principios de la evolución natural: reproducción mediante operadores genéticos y selección de los mejores individuos, correspondiendo éstos a las mejores soluciones del problema a optimizar.

Para aprovechar las ventajas del modelo del proceso evolutivo en la resolución de un problema de optimización, se deben establecer las siguientes correspondencias:

1. Una apropiada codificación de las posibles soluciones del problema representará a éstas de la misma forma que el cromosoma. Representa a los individuos de la especie. Dada esta unívoca relación, se usarán indistintamente los términos solución, codificación, cromosoma o individuo.
2. La adecuación de cada solución será una medida del comportamiento de ésta en el problema particular considerado. Normalmente, es el valor objetivo de la solución. Así, una solución está más adecuada a un problema cuanto mejor sea su valor objetivo.
3. Se definirán unos operadores genéticos que, al actuar sobre una o varias soluciones, suministren una o más soluciones al alterar genéticamente los cromosomas. Juegan el papel del cruce y la mutación en el proceso evolutivo natural.

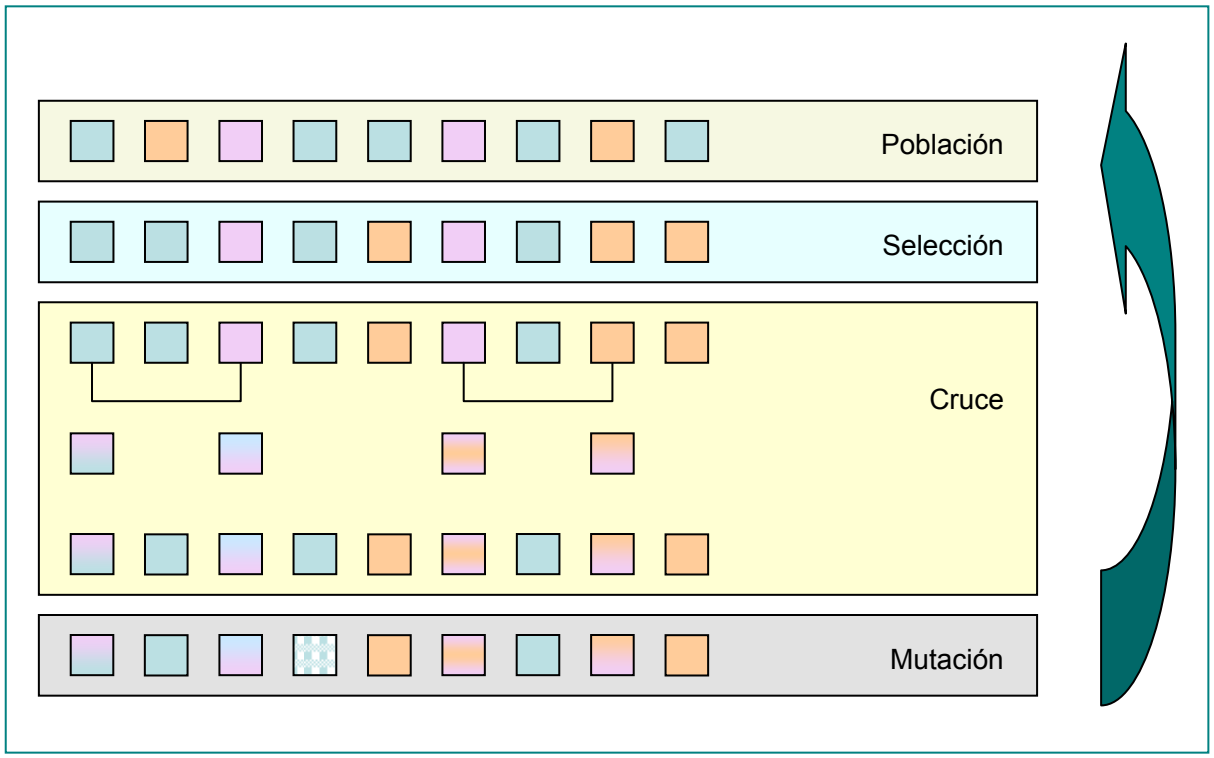


Figura 5. Esquema de funcionamiento de un algoritmo genético

La versión *simple* o *canónica* del algoritmo genético trabaja siguiendo los siguientes pasos:

1. Generar una población inicial de soluciones.
2. Seleccionar, de la población actual, las soluciones mejor adaptadas.
3. Cruzar algunas soluciones para obtener su descendencia.
4. Mutar algunas soluciones para obtener las soluciones mutadas.
5. Elegir las soluciones que sobreviven y formarán la nueva generación.
6. Si no se alcanza el criterio de parada volver al paso 2.

Al finalizar los pasos anteriores, la mejor solución de la población es la que se propone como solución del problema.

3.1. Un ejemplo

Planteemos el siguiente problema de optimización:

> Encontrar el máximo global de la función real de una variable $f(x) = -x^2 + 5x + 3$ sobre el intervalo (0,4) con una precisión de 3 dígitos tras el punto decimal.

La función f es un polinomio de segundo grado cuyo máximo absoluto se encuentra en $x = 2{,}5$ alcanzo un valor de 9,25 como es fácil de comprobar. La forma de esta función en el intervalo dado se refleja en la figura siguiente.

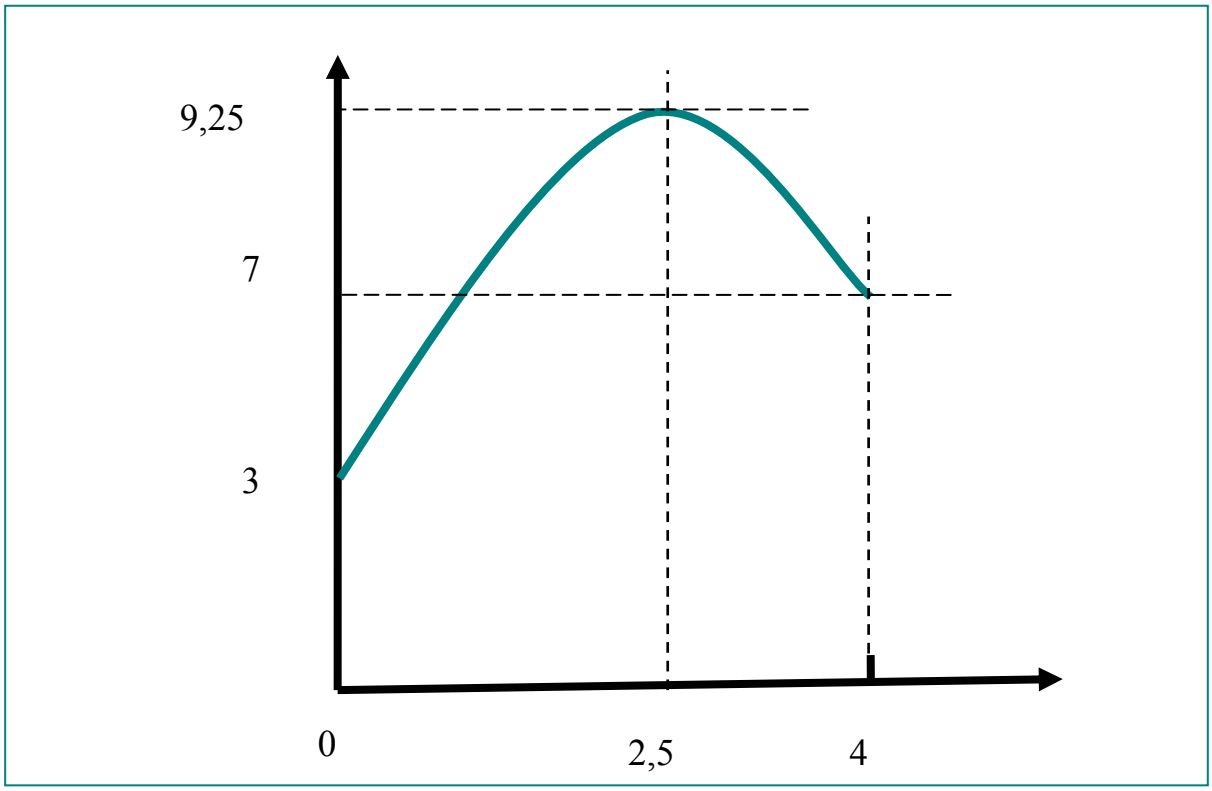


Figura 6. Gráfica de la función a optimizar

Para representar de forma apropiada las soluciones de este problema usamos la representación binaria. Representamos cada número real del intervalo (0,4) en base 2 con una precisión de 10^{-3}. Puesto que $4 = 100_2$ y $0{,}0000000001_2 = 2^{-10} < 10^{-3}$ será suficiente con tomar como soluciones posibles del problema a los números reales representados en forma binaria por dos cifras antes de la coma y 10 detrás. Algunos ejemplos se muestran a continuación.

Cromosoma	Número real
[1 1,1 1 0 0 1 0 1 0 1 0]	3,791015625
[0 1,1 0 0 0 1 1 1 0 1 0]	1,546640625
[0 0,0 0 0 1 1 1 1 0 1 0]	0,119140600
[1 0,0 1 0 1 1 1 0 0 0 0]	2,359375000
[1 0,0 0 0 0 1 1 1 0 1 0]	2,056640600

Tabla 1. Población inicial

El algoritmo genético empieza con una población de soluciones del problema de un tamaño determinado. Una forma sencilla de construir una población inicial de tamaño n consiste en generar sus soluciones aleatoriamente. En este caso, se trata de obtener al azar n cadenas binarias de tamaño 12. En la tabla siguiente se muestra una de tales poblaciones iniciales con $n = 5$ individuos. En dicha tabla se recoge la cadena binaria, el número real que representa y el valor de dicha cadena.

Cromosoma	Número real: x_i	Valor de la función: $f(x_i)$
$x_1 =$ [1 1,1 1 0 0 1 0 1 0 1 0]	$x_1 = 3{,}791015625$	$f(x_1) = 5{,}8865$
$x_2 =$ [0 1,1 0 0 0 1 1 1 0 1 0]	$x_2 = 1{,}546640625$	$f(x_2) = 8{,}3411$
$x_3 =$ [0 0,0 0 0 1 1 1 1 0 1 0]	$x_3 = 0{,}119140600$	$f(x_3) = 4{,}9638$
$x_4 =$ [1 0,0 1 0 1 1 1 0 0 0 0]	$x_4 = 2{,}359375000$	$f(x_4) = 9{,}2302$
$x_5 =$ [1 0,0 0 0 0 1 1 1 0 1 0]	$x_5 = 2{,}056640600$	$f(x_5) = 9{,}0534$

Tabla 2. Evaluación de la población inicial

El mayor valor del objetivo en la población es $f(x_3) = 9{,}2302$ y el valor promedio es 7,495.

El siguiente paso será seleccionar de la población los elementos que van a someterse al cruzamiento y eventualmente a la mutación. El procedimiento de selección debe ser tal que los individuos más adaptados, es decir, las soluciones, x, con mayor valor de la función $f(x)$, tengan mayor probabilidad de ser elegidos. Las probabilidades proporcionales al valor de la función se obtienen dividiendo cada $f(x_i)$ por la suma de todas ellas; en nuestro ejemplo es 37,475.

Cromosoma	Valor de la función: $f(x_i)$	Probabilidades
$x_1 =$ [1 1,1 1 0 0 1 0 1 0 1 0]	$f(x_1) = 5{,}8865$	$f(x_1) / \sum f(x_i) = 0{,}157$
$x_2 =$ [0 1,1 0 0 0 1 1 1 0 1 0]	$f(x_2) = 8{,}3411$	$f(x_2) / \sum f(x_i) = 0{,}223$
$x_3 =$ [0 0,0 0 0 1 1 1 1 0 1 0]	$f(x_3) = 4{,}9638$	$f(x_3) / \sum f(x_i) = 0{,}132$
$x_4 =$ [1 0,0 1 0 1 1 1 0 0 0 0]	$f(x_4) = 9{,}2302$	$f(x_4) / \sum f(x_i) = 0{,}246$
$x_5 =$ [1 0,0 0 0 0 1 1 1 0 1 0]	$f(x_5) = 9{,}0534$	$f(x_5) / \sum f(x_i) = 0{,}241$

$\sum f(x_i) = 37{,}475$

Tabla 3. Probabilidades de selección

Una de las técnicas más usuales para ello es emplear el conocido como método de la rueda de hilar. La idea es construir una rueda de ruleta con sectores proporcionales a estas probabilidades.

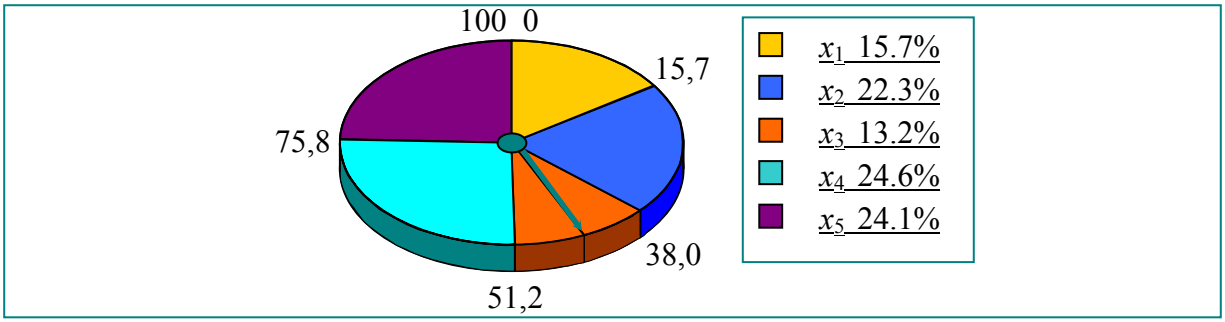


Figura 7. Método de la Rueda de la Ruleta

En términos matemáticos se asigna a cada solución, x, de la población un segmento del intervalo [0,1] proporcional a su probabilidad; es decir al valor relativo de la función f:

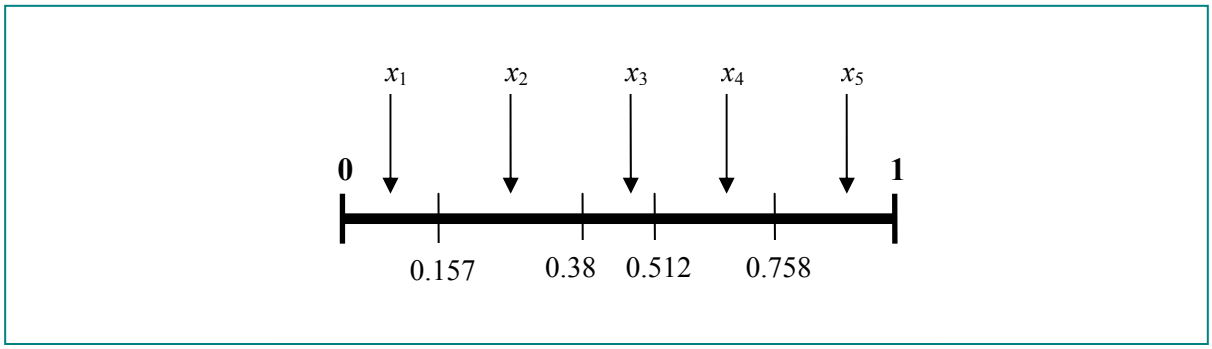


Figura 8. Método de la Rueda de la Ruleta sobre el intervalo [0,1]

Si se obtienen los números aleatorios 0,532 – 0,776 – 0,723 – 0,285 – 0,846 resultan seleccionados los individuos x_4, x_5, x_4, x_2 y x_5.

Cromosoma	**Número real:** x_i	**Valor de la función:** $f(x_i)$
x_4 = [10,0101110000]	x_4 = 2,359375000	$f(x_4)$ = 9,2302
x_5 = [10,0000111010]	x_5 = 2,056640600	$f(x_5)$ = 9,0534
x_4 = [10,0101110000]	x_4 = 2,359375000	$f(x_4)$ = 9,2302
x_2 = [01,1000111010]	x_2 = 1,546640625	$f(x_2)$ = 8,3411
x_5 = [10,0000111010]	x_5 = 2,056640600	$f(x_5)$ = 9,0534

Tabla 4. Individuos seleccionados de la población

Nótese que al seleccionar con mayor probabilidad los mejores individuos la media de los valores seleccionados es mejor que la media de la población de la que proviene. En este caso, el promedio de los valores de los individuos seleccionados sube hasta 8,58146.

El siguiente paso consiste en obtener parejas de estos individuos para cruzarlos y aplicarles el operador de cruce considerado. Uno de los operadores más usuales para cadenas binarias consiste en seleccionar al azar una posición de cruce e intercambiar las subcadenas de la derecha.

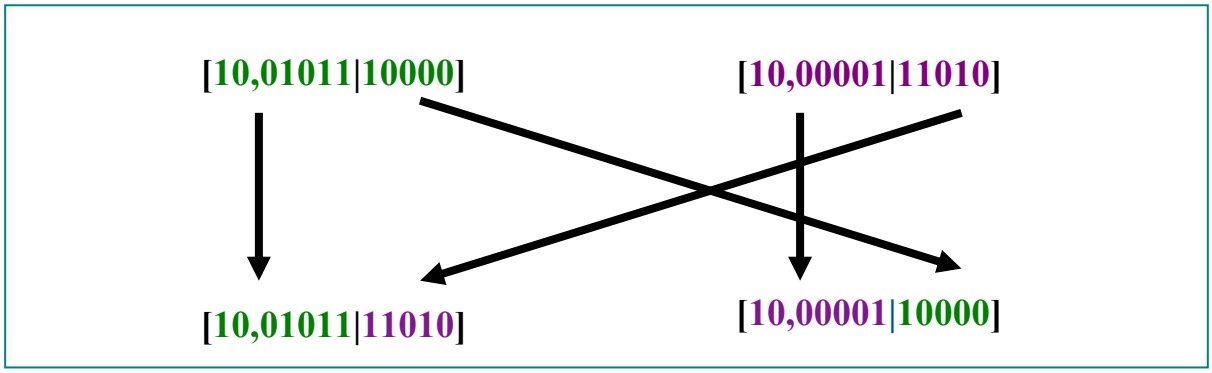


Figura 9. Cruzamiento unipuntual

Por tanto de los padres x_4 = [10,0101110000] y x_5 = [10,0000111010] obtenemos los dos hijos [10,0101111010] y [10,0000110000] que si sustituyen a sus padres tenemos una nueva población. Por tanto, de la población inicial

Cromosoma	**Número real:** x_i	**Valor de la función:** $f(x_i)$
x_1 = [1 1,1 1 0 0 1 0 1 0 1 0]	x_1 = 3,791015625	$f(x_1)$ = 5,8865
x_2 = [0 1,1 0 0 0 1 1 1 0 1 0]	x_2 = 1,546640625	$f(x_2)$ = 8,3411
x_3 = [0 0,0 0 0 1 1 1 1 0 1 0]	x_3 = 0,119140600	$f(x_3)$ = 4,9638
x_4 = [1 0,0 1 0 1 1 1 0 0 0 0]	x_4 = 2,359375000	$f(x_4)$ = 9,2302
x_5 = [1 0,0 0 0 0 1 1 1 0 1 0]	x_5 = 2,056640600	$f(x_5)$ = 9,0534

Tabla 5. Población inicial

con mejor valor 9,2302 y promedio 7,495 pasamos a la población tras el cruzamiento y sustitución

Cromosoma	**Número real:** x_i	**Valor de la función:** $f(x_i)$
x_1 = [1 1,1 1 0 0 1 0 1 0 1 0]	x_1 = 3,791015625	$f(x_1)$ = 5,8865
x_2 = [0 1,1 0 0 0 1 1 1 0 1 0]	x_2 = 1,546640625	$f(x_2)$ = 8,3411
x_3 = [0 0,0 0 0 1 1 1 1 0 1 0]	x_3 = 0,119140600	$f(x_3)$ = 4,9638
x'_4 = [1 0,0 1 0 1 1 1 1 0 1 0]	x'_4 = 2,369140625	$f(x'_4)$ = 9,2329
x'_5 = [1 0,0 0 0 0 1 1 0 0 0 0]	x'_5 = 2,046875000	$f(x'_5)$ = 9,0447

Tabla 6. Población tras cruzamiento y sustitución

con mejor valor 9,2329 y promedio 7,483.

Pero si los nuevos individuos generados sustituyen a los dos peores de la población obtenemos la población

Cromosoma	Número real: x_i	Valor de la función: $f(x_i)$
x'_4 = [1 0,0 1 0 1 1 1 1 0 1 0]	x'_4 = 2,369140625	$f(x'_4)$ = 9,2329
x_2 = [0 1,1 0 0 0 1 1 1 0 1 0]	x_2 = 1,546640625	$f(x_2)$ = 8,3411
x'_5 = [1 0,0 0 0 0 1 1 0 0 0 0]	x'_5 = 2,046875000	$f(x'_5)$ = 9,0447
x_4 = [1 0,0 1 0 1 1 1 0 0 0 0]	x_4 = 2,359375000	$f(x_4)$ = 9,2302
x_5 = [1 0,0 0 0 0 1 1 1 0 1 0]	x_5 = 2,056640600	$f(x_5)$ = 9,0534

Tabla 6. Población tras cruzamiento y sustitución por los peores

que tiene promedio 8,98046.

Además del cruzamiento, el otro operador básico de los algoritmos genéticos es la mutación, que esencialmente modifica al azar algunos de los genes del cromosoma del individuo que muta. Para cadenas binarias la modificación del gen consiste en intercambiar uno de los posibles valores por el otro; 0 por 1 o 1 por 0. La probabilidad de mutación, y por tanto la frecuencia con la que se aplica, suele ser muy pequeña con respecto al cruzamiento. Para aplicarlo a un individuo concreto, una vez fijada la probabilidad de mutación, se puede decidir, gen a gen, si muta o no. Otra posibilidad es decidir para cada individuo si muta o no con esa probabilidad, y en caso de mutar un individuo, se elige al azar el gen que muta y se modifica su valor.

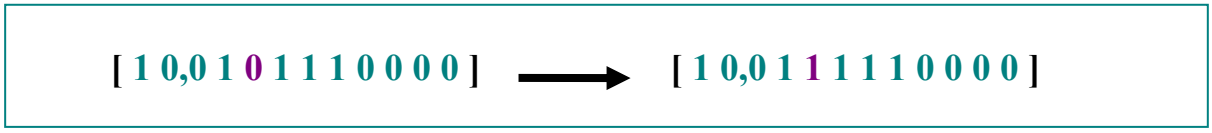


Figura 9. Mutación en el quinto gen

Si en nuestro ejemplo muta el elemento x_4 = [10,0101110000] en la posición 5 se obtiene el nuevo cromosoma x''_4 = [10,01**1**1110000] que corresponde a 2,484375 y llega a alcanzar el valor del objetivo 9,2497.

De esta forma la población, tras el primer cruzamiento y la primera mutación queda

Cromosoma	Número real: x_i	Valor de la función $f(x_i)$
x'_4 = [1 0,0 1 0 1 1 1 1 0 1 0]	x'_4 = 2,369140625	$f(x'_4)$ = 9,2329
x_2 = [0 1,1 0 0 0 1 1 1 0 1 0]	x_2 = 1,546640625	$f(x_2)$ = 8,3411
x'_5 = [1 0,0 0 0 0 1 1 0 0 0 0]	x'_5 = 2,046875000	$f(x'_5)$ = 9,0447
x''_4 = [1 0,0 1 1 1 1 1 0 0 0 0]	x''_4 = 2,484375000	$f(x''_4)$ = 9,2497
x_5 = [1 0,0 0 0 0 1 1 1 0 1 0]	x_5 = 2,056640600	$f(x_5)$ = 9,0534

Tabla 7. Población tras cruzamiento y sustitución por los peores

En esta situación se ha concluido el primer ciclo evolutivo y la solución propuesta es la mejor solución de la población que se trata de la solución x^* = 2,484375 que corresponde al cromosoma [10,0111110000] y tiene un valor objetivo $f(x^*)$ = 9,2497. La solución aportada está muy cerca de la verdadera solución optima x^* = 2,5 que corresponde al cromosoma [10,1000000000] con un valor objetivo $f(x^*)$ = 9,25.

De esta forma se cierra un ciclo evolutivo y se vuelve otra vez a trabajar con la nueva población. Este proceso se itera hasta que se cumpla un criterio de parada.

3.2. La metáfora

Los algoritmos genéticos se pueden utilizar para resolver prácticamente cualquier tipo de problema de optimización. Para ello es necesario establecer la correspondencia entre los distintos elementos del problema y las componentes del algoritmo genético. Esta correspondencia se deriva de la particular interpretación de la metáfora que inspira los algoritmos genéticos que viene reflejada en la tabla siguiente.

Evolución Natural	Algoritmo Genético
Evolución	Heurística
Ambiente	Problema
Población	Conjunto
Individuo	Solución
Adaptación	Calidad
Cromosoma	Representación
Mutación	Movimiento
Cruzamiento	Combinación

Tabla 8. Metáfora de aplicación de los algoritmos genéticos

Una parte fundamental del diseño e implementación de un algoritmo genético para resolver un problema consiste en hacer una interpretación adecuada de esta metáfora con los matices pertinentes. El primer paralelismo presente en la metáfora se presenta entre la propia *evolución* como un proceso natural y el algoritmo genético como una *heurística* que guía el proceso de convertir un conjunto arbitrario de soluciones del problema en uno que contenga la solución óptima, o una próxima a serlo. El *ambiente* natural en el que se desarrolla el fenómeno de la evolución y las características del *problema* abordado por el algoritmo genético representa el conjunto de característica que condiciona la propia evolución. El objeto de interés es la *población* de individuos que evoluciona en la naturaleza y los elementos de un *conjunto* de soluciones que se van actualizando continuamente. Sus miembros se denominan, respectivamente, *individuos* y *soluciones*. En ellos se mide o evalúa la *adaptación* al ambiente y su *calidad* como soluciones del problema para determinar el papel que juegan en la evolución conjunta y el grado de éxito alcanzado. Los *cromosomas* reflejan la manera de almacenar la información necesaria de los individuos para determinar las características de interés de la misma forma que la *representación* de las soluciones determinan las características que se consideran importantes para el problema. Finalmente, las operaciones que permiten la interacción entre los individuos o soluciones y los cambios que se producen en ellos son la *mutación* o *movimiento* en el espacio de soluciones y el *cruzamiento* o *combinación* de soluciones.

Una vez establecida la interpretación de la metáfora que da lugar al diseño del algoritmo genético existe una gran variedad de formas de implementar las distintas componentes que lo constituyen. Una de las versiones más simples, que generalmente se denomina Algoritmo Genético Canónico, es la descrita en el pseudocódigo siguiente:

```
Procedure Algoritmo Genético
     Begin
          t ← 0 ;
          Inicializar P(t) ;
          Evaluar P(t) ;
          Repeat
               Seleccionar P(t+1) desde P(t) ;
               Operar P(t+1) ;
               Evaluar P(t+1) ;
               t ← t + 1 ;
          Until Criterio_de_Parada ;
     End ;
```

Figura 10. Pseudocódigo de un algoritmo genético estándar

El algoritmo genético canónico sigue el esquema:

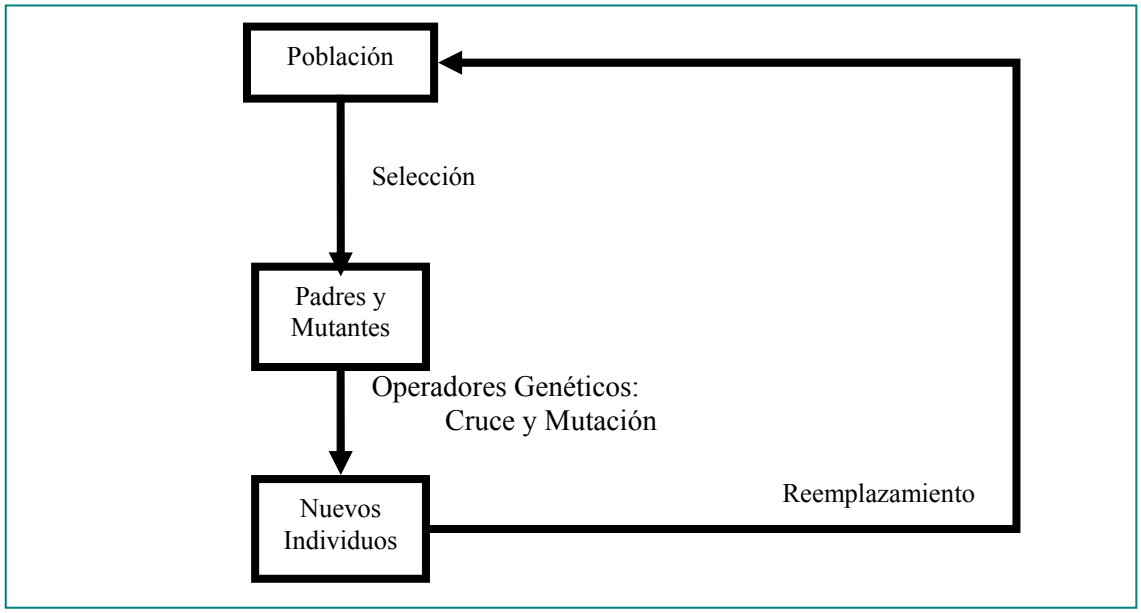


Figura 11. Esquema de funcionamiento de un algoritmo genético estándar

3.3 Variantes fundamentales

Las diferentes versiones de los algoritmos genéticos surgen al especificar los distintos detalles de las componentes de un algoritmo genético. Dos elementos importantes comunes a la mayoría de los algoritmos genéticos son el *tamaño de la población* y el *criterio de parada*.

El tamaño de la población suele ser de varios cientos o miles de individuos y su elección se realiza a partir de la propia experiencia tratando de buscar un equilibrio entre el esfuerzo de cómputo y la capacidad de barrer completamente el espacio de búsqueda. El criterio de parada puede ser cualquiera de los comúnmente utilizados en los métodos de búsqueda heurística referidos al esfuerzo de cómputo o al estancamiento de la búsqueda. El esfuerzo de cómputo se puede medir a través del

tiempo de CPU o número de operaciones básicas aplicadas (como por ejemplo, número de cruzamientos), aunque lo más corriente es limitar el número de generaciones o ciclos evolutivos que se ejecutan. El tamaño de la población y el número de generaciones dan lugar al número total de individuos que se evalúan y su combinación se utiliza para establecer comparaciones experimentales en el rendimiento de distintas estrategias implementadas.

El *estancamiento* de la población, en el sentido de que no mejora la evaluación media o la mejor solución aportada, en un cierto número de etapas es un criterio de parada razonable porque indica claramente que esfuerzos adicionales no producirán mejoras apreciables. Sin embargo, los algoritmos genéticos tienen el riesgo de estancarse antes de alcanzar soluciones de alta calidad produciendo una convergencia prematura, por lo que se dotan de instrumentos para evitarlo, frecuentemente basados en el control de la *diversidad de la población*. Esta diversidad se mide generalmente en términos de los valores promedio o mejor de la función objetivo en la población, aunque algunas propuestas más avanzadas tienen en cuenta otros aspectos de diferenciación entre sus individuos.

Existen dos tipos de modelos clásicos

Modelo Generacional: En cada iteración, a partir de la población existente se genera una población completa de nuevos individuos con los operadores genéticos. La nueva población reemplaza a la anterior.

Modelo Estacionario: En cada iteración, se seleccionan dos padres de la población y se les aplican los operadores genéticos obteniendo dos hijos (posiblemente mutados). Los nuevos individuos reemplazan a sus padres.

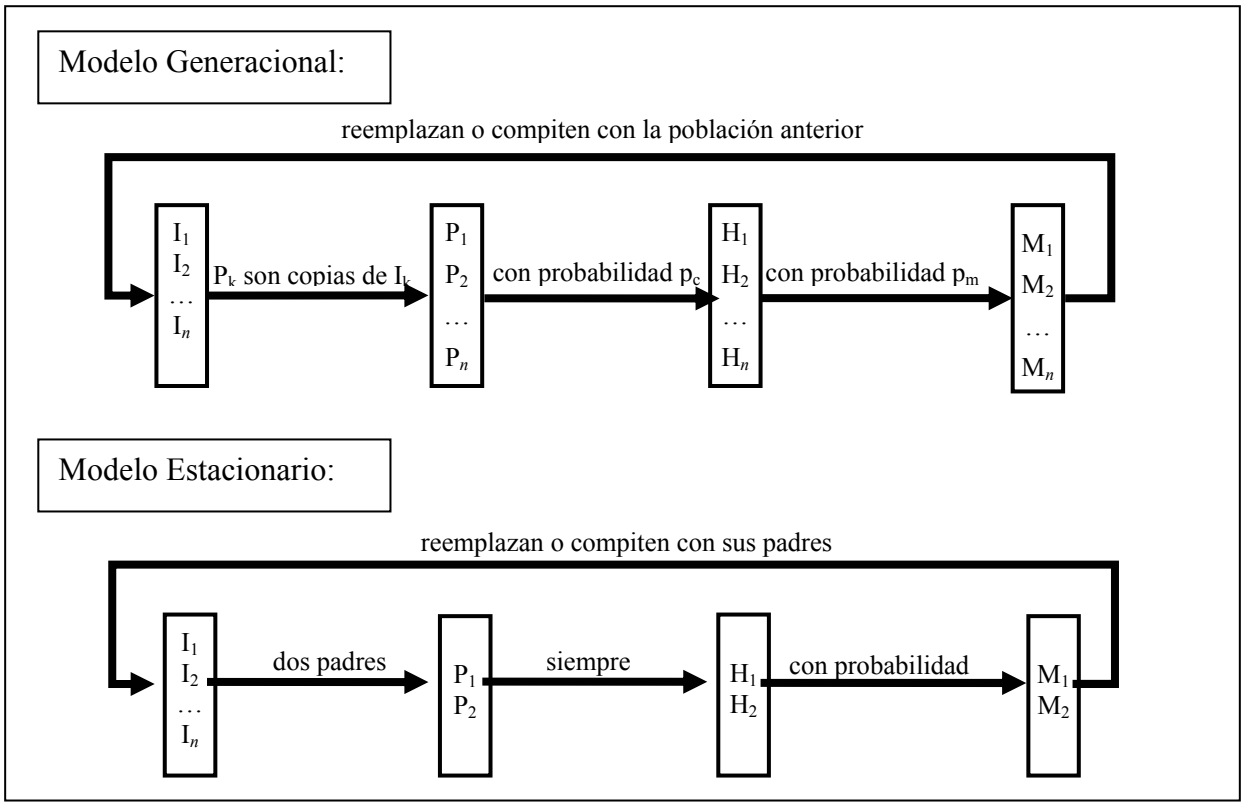


Figura 12. Modelos generacional y estacional de un algoritmo genético

El *elitismo* persigue que se mantengan en la población los mejores individuos que sólo pueden ser reemplazados si aparece un individuo mejor.

La *competición* utiliza una comparación entre los nuevos individuos y los existentes para establecer cuál permanece en la población y cuál desaparece. El mejor reemplaza siempre al peor o con una probabilidad menor que 1, pero mayor que 1/2; regulando la presión selectiva.

Codificación.

Tradicionalmente, y debido a las primeras aplicaciones de los algoritmos genéticos para resolver problemas con variables numéricas continuas, se ha usado como técnica de codificación la binaria (las variables reales se representan con *cadenas binarias* de ceros y unos). Sobre las cadenas se han definido de forma sencilla multitud de operadores. Esta sencillez de definición, y el hecho de que casi la totalidad del estudio teórico sobre el comportamiento del algoritmo se haya basado en esta codificación, ha motivado que muchos investigadores usen la codificación binaria para representar las soluciones de sus problemas. Sin embargo, en muchos problemas no es fácil codificar las soluciones a través de cadenas binarias; o no es lo más eficiente y eficaz. Además si el problema tiene restricciones en las variables, puede ser complicado trabajar con operadores genéticos que puedan dar lugar a soluciones no factibles.

Para representar variables reales acotadas, se utiliza la siguiente técnica. Si queremos utilizar una variable x con precisión ε y rango de variación $[a,b]$ hay que usar una cadena binaria con tamaño $c = 1 + \lfloor \log 2\ (b-a) \rfloor / \varepsilon$. La cadena que representa al valor v será la expresión en base 2 del número $x = 2^c\ (v - a)/(b - a)$. Recíprocamente, el valor v correspondiente a una cadena binaria que en base 2 corresponda al número x es $v = a + x \cdot 2^{-c}\ (b - a)$. Por ejemplo, si queremos representar una variable x en el intervalo [1,3] con una precisión de $\varepsilon = 10{-}6$ necesitamos una cadena de tamaño $c = 20$. Así, por ejemplo, la cadena binaria [10010111000001111100101], es la expresión del número 1237477 en base 2 que corresponde al valor $v = 1 + 1237477\ (3 - 1)\ 2^{-20} = 2{,}180150$.

Operadores de cruce.

Algunos de los operadores básicos más usuales son los siguientes.

- *Cruce unipuntual*: se selecciona al azar una posición de cruce y se intercambian las subcadenas situadas a la derecha de esta posición.

Padres	Hijos
[1 0 0 1 0 1 1|1 0 0 0 0]	[1 0 0 1 0 1 1|1 1 0 1 0]
[0 1 1 0 0 0 1|1 1 0 1 0]	[0 1 1 0 0 0 1|1 0 0 0 0]

Tabla 9. Operador de Cruce unipuntual

- *Cruce bi-puntual*: se selecciona al azar dos posiciones de cruce y se intercambian las subcadenas situadas entre dichas posiciones

Padres	Hijos
[1 0 0 1|0 1 1 1 0|0 0 0]	[1 0 0 1|0 0 1 1 1|0 0 0]
[0 1 1 0|0 0 1 1 1|0 1 0]	[0 1 1 0|0 1 1 1 0|0 1 0]

Tabla 11. Operador de Cruce bi-puntual

- *Cruce aleatorio*: se decide al azar si se intercambia o no cada elemento de las subcadenas.

Padres	Hijos
[1 0 0 1 0 1 1 1 0 0 0 0]	[1 0 1 1 0 1 1 1 1 0 0 0]
[0 1 1 0 0 0 1 1 1 0 1 0]	[0 1 0 0 0 0 1 1 0 0 1 0]

Esquemas de selección.

Para elegir los 2p padres, p parejas, con los que realizar los cruzamientos se puede aplicar, entre otras, una de las siguientes fórmulas:

- *Por Torneo*: Se eligen k al azar y de ellos se toma el mejor; se repite 2p veces como haga falta.
- *Orden lineal*: Se ordenan de mejor a peor y se escogen los 2p mejores.
- *Aleatoria pura*: Se escogen totalmente al azar 2p individuos.
- *Procedimiento de la ruleta*: se asignan probabilidades y se eligen 2p padres de acuerdo a ellas.
- *Diverso/aleatorio*: Se escoge p veces un padre al azar que se empareja con el individuo más diferente a él.
- *Diverso/elitista*: Se escogen los p mejores y cada uno de ellos se empareja con el individuo más diferente a él.

Además, se pueden combinar varios de estos métodos, lo que es bastante corriente en aplicaciones prácticas exitosas.

Modelos con probabilidades uniformes.

Para el estudio de las propiedades teóricas del comportamiento de los algoritmos genéticos hay que acudir a modelos simplificados, como el algoritmo genético con *probabilidades uniformes*. En esta versión del algoritmo genético, se considera un sólo operador cruce y uno de mutación, organizando la aflicción de los operadores genéticos a los elementos seleccionados de la población en una fase de cruce y seguida de una de mutación. En la fase de cruce, para cada individuo seleccionado de la población, se decide, con una probabilidad p_c, si ésta se toma como padre de un cruce. A continuación se aparean aleatoriamente las codificaciones seleccionadas y se aplica el operador cruce a cada una de estas parejas. Las dos codificaciones hijas resultantes reemplazan a sus padres. Durante la fase de mutación, para cada codificación seleccionada de la población, y para cada uno de los símbolos que la forman, se decide con una probabilidad dada, p_m, si éste se altera o no. Las codificaciones mutadas ocupan el lugar de aquellas de las que provienen.

3.4. Algoritmo Genético para problemas combinatorios

La codificación binaria de variables de cualquier tipo permite teóricamente aplicar un algoritmo genético a cualquier problema. De hecho, la forma en que se almacenan los datos a bajo nivel en el ordenador son siempre cadenas de ceros y unos. Sin embargo, existen otras alternativas más especializadas de la codificación adoptada y del diseño e implementación de los diferentes operadores que son esenciales en el éxito práctico de algoritmos genéticos a problemas reales. Para mostrar algunas de las cuestiones específicas de la aplicación de algoritmos genéticos a problemas combinatorios escogemos un tipo importante de problema de optimización combinatoria: *los problemas de selección*. Estos problemas combinatorios con aplicaciones relevantes en clasificación y en bioinformática, son los consistentes en seleccionar de forma óptima un número fijo p de elementos de un universo de n miembros. En particular consideramos *el problema de la p-mediana,* un problema

muy estudiado en la literatura y con importantes aplicaciones en planificación logística que permite ilustrar cómo utilizar algoritmos genéticos para resolver problemas combinatorios. Una bibliografía actualizada sobre este problema puede verse en Reese (2006) y sobre la aplicación de metaheurística en Mladenović et al. (2007); en particular, sobre la aplicación de algoritmos genéticos Alp et a. (2003). Vamos a describir los elementos para diseñar un Algoritmo Genético para estos problemas: Codificación, Selección, Cruzamaniento, Mutación,…

El problema de la *p*-mediana se formula en los siguientes términos: "Dado un conjunto finito U de m puntos de demanda y un conjunto L de n puntos de localización, se desean seleccionar p de estos puntos, llamadas *medianas* que hagan mínima la suma de las distancias a los puntos de U".

$$\min_{\substack{|X|=p \\ X \in L}} \sum_{u \in U} \min_{x \in X} Dist(u,x)$$

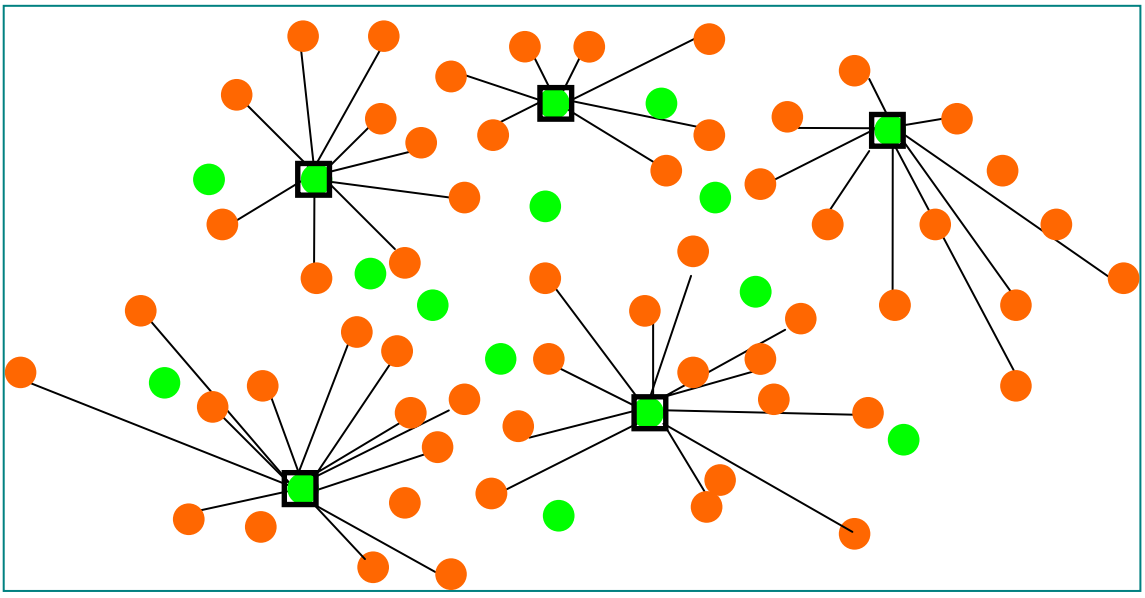
El problema de la *p*-mediana pertenece a una importante clase de problemas de localización que cuenta en general con los siguientes elementos:

- Un conjunto de localizaciones potenciales:
  $L = \{ v_1 , v_2 , \dots , v_m \}$.
- Un conjunto de usuarios del servicio.
  $U = \{ u_1 , u_2 , \dots , u_n \}$.
- Una función $f(X)$ a minimizar definidas para cualquier $X \subseteq$ L con $|X| = p$.
- La función f se define a partir de las distancias usuario-localización:
  $Dist: U \times L \rightarrow R$

Entre estos problemas, los más comunes son los de asignación directa donde cada usuario se asigna al punto de localización seleccionado más cercano y la función objetivo de un conjunto de localizaciones X se expresa en términos de las distancias:

$$Dist(u_i ,X) = \min \{Dist(u_i ,v_j) : v_j \in X \} \rightarrow f(X)$$

Algunos problemas clásicos de este tipo son la *p*-mediana, el *p*-centro y el λ-centdian.



Codificación binaria.

Dado que el conjunto de posibles selecciones es finito, se supone ordenado o indexado, y cada solución del problema se representa por una n-upla binaria con p unos; un 1 en la i-ésima posición indica que el i-ésimo item está en la solución. Las principales ventajas de esta codificación son que pueden utilizarse los operadores genéticos empleados por el algoritmo genético clásico. Los inconvenientes son que esos operadores suministran, en general, soluciones no factibles, y que las $n-p$ posiciones en las que aparece un cero no aportan información adicional y, sin embargo, ocupan memoria. Para subsanar el primero de los inconvenientes se puede penalizar la función objetivo con la desviación entre el valor de p y el de unos en la solución.

Ejemplo: Si hay $n = 7$ items y se eligen $p = 4$ de ellos para cada solución, algunos ejemplos de codificaciones de soluciones son: [1 0 1 0 0 1 1], [1 0 0 0 1 1 1], [1 1 1 1 0 0 0], …

Codificación indexal.

Otra codificación es la p-upla ordenada de los índices de los items seleccionados. La ordenación se utiliza para asegurar que en las mismas circunstancias, el operador cruce, que se define a continuación, suministra las mismas soluciones hijas. Se llama a esta codificación, indexal.

Ejemplo: Si el número de items posibles es $n = 7$ y se pretenden localizar $p = 5$ servicios, la solución [1 0 1 0 1 1 1] se codifica como [1 3 5 6 7].

La codificación indexal ha mostrado un mayor rendimiento que la codificación binaria en algunos problemas prácticos.

Cruce indexal.

El operador de cruce indexal intercambia los índices a la derecha de la posición de cruce. Esta operación se diseña de forma que cada vez que actúe sobre dos codificaciones padres con el mismo punto de cruce, dé lugar a las mismas codificaciones hijas, y asegurando que las codificaciones resultantes sean factibles. Lo primero se consigue utilizando una ordenación de los índices, y lo segundo marcando aquellos índices que necesariamente deben pasar de una codificación a otra para que se cumpla el requerimiento de factibilidad. El cruce se implementa eligiendo al azar una de las $p-1$ posiciones de ruptura. Las posiciones situadas a la derecha de ésta, son las posiciones de cruce.

El operador de cruce indexal mantiene, en cada una de las codificaciones hijas, los índices de los respectivos padres situados a la izquierda de la posición de cruce. En efecto, debido a las ordenaciones usadas para codificar las soluciones y aplicar el cruce, los índices marcados ocupan la misma posición en cada una de las soluciones. Si un índice a la izquierda de la posición de cruce no es marcado, no altera su posición dentro de la solución y no abandona ésta durante el intercambio de índices. Si ese índice es marcado pueden ocurrir dos cosas:

1) tras las ordenaciones correspondientes, dicho índice mantiene su posición a la izquierda del punto de cruce, o
2) altera su posición situándose a la derecha de dicho punto.

En ambos casos, tras el cruce, forma parte de la nueva solución.

El procedimiento para implementar el cruce indexal es el siguiente. Para cada una de las codificaciones de la pareja desarrollar los siguientes pasos:

- Marcar los índices que aparezcan en las posiciones de cruce de la codificación enfrentada;
- Marcar los índices de las posiciones de cruce que están presentes en la otra codificación;
- Colocar en las primeras posiciones, y en orden creciente, los índices no marcados; colocar en las últimas posiciones, y en orden creciente, los índices marcados;
- Intercambiar los índices de las posiciones de cruce;
- Eliminar las marcas y ordenar las codificaciones resultantes.

Ejemplo: Supongamos que las codificaciones padres son: [1 3 8 9 10] y [7 9 10 11 15]. Si la posición de cruce es 2, las marcas son las siguientes: [1 3 8 9* 10*] y [7 9* 10* 11 15]. Tras ordenar los índices no marcados quedan las secuencias: [1 3 8 9* 10*] y [7 11 15 9* 10*]. Aplicando la operación de cruce resultan: [1 3 15 9 10] y [7 11 8 9 10], que una vez ordenadas dan lugar a las codificaciones hijas: [1 3 9 10 15] y [7 8 9 10 11].

La mutación indexal se define de forma análoga a la considerada para cadenas binarias, esto es, cambia el índice a mutar por otro no presente en la solución.

Ejemplo: Si la solución actual es [1 3 8 9 10] y el índice a mutar es el 3, una de las soluciones resultantes puede ser [1 5 8 9 10].

4. Conclusiones

Los algoritmos genéticos constituyen una de los instrumentos más relevantes de las técnicas de Inteligencia Artificial inspiradas en la naturaleza. Su interés teórico y práctico está sustentado en estudios e investigaciones ampliamente difundidas en publicaciones de alto prestigio. Desde el punto de vista teórico muestran cómo la naturaleza proporciona modelos sencillos que permiten dar solución a problemas importantes en la sociedad actual. Desde el punto de vista práctico revelan cómo obtener fácilmente algoritmos eficientes que tengan éxito en importantes aplicaciones prácticas. En este trabajo mostramos como se aplican los fundamentos de los algoritmos genéticos, tanto para los problemas de optimización global para los que originalmente fueron concebidos, como para los problemas combinatorios que con cada vez mayor relevancia práctica aparecen en contexto actuales de la ciencia, la ingeniería, la gestión o la actividad empresarial e industrial.

Bibliografía

Alp, O., Erkut, E., Drezner, Z. (2003). An Efficient Genetic Algorithm for the p-Median Problem. *Annals of Operations Research*, 122 (1-4), 21-42.

Darwin, C. (1859). *On the Origin of Species by Means of Natural Selection*. Murray, Londres.

Goldberg, D., Sastry, K. (2008) *Genetic Algorithms. The Design of Innovation*. Springer, Berlín.

Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, MA.

Mladenović, N., Brimberg, J., Hansen, P., Moreno-Pérez J.A. (2007). The p-median problem: A survey of metaheuristic approaches. *European Journal of Operational Research*, 179(3), 927-939.

Reese, J. (2006). Solution methods for the p-median problem: An annotated bibliography. *Networks*, 48 (3), 125-142.

Belén Melián Batista, Departamento de Estadística, I.O. y Computación, Universidad de La Laguna, 38271 La Laguna. Nació el 11 de julio de 1976 en Arucas, Las Palmas de Gran Canaria. Es Doctora en Informática y Profesora Titular de la Universidad de La Laguna. Ha sido miembro del comité de programa, comité organizador y revisora de varios congresos de Inteligencia Artificial y Heurísticas para la resolución de problemas. Es coautora de artículos de investigación publicados en revistas internacionales como IEEE Transactions on Fuzzy Systems, Journal of Heuristics, European Journal of Operational Research, Networks, Computers and Operations Research, Parallel Computing, actuando como editora invitada en algunas de ellas.

José Andrés Moreno Pérez, Departamento de Estadística, I.O. y Computación, Universidad de La Laguna, 38271 La Laguna. Nació el 14 de mayo de 1958 en La Laguna, Tenerife. Licenciado en Matemáticas en 1980 por la Universidad Complutense de Madrid, con el mejor expediente de su promoción tanto en la especialidad de Estadística como en la de Investigación Operativa, y doctor en Matemáticas por la misma universidad con premio extraordinario de doctorado. Ha sido profesor de la Universidad Complutense hasta 1986 y en la actualidad catedrático de Ciencias de la Computación e Inteligencia Artificial. Es coautor de más de 50 artículos de investigación publicados en revistas internacionales especializadas en heurísticas y actuado como editor invitado en varias de ellas.

José Marcos Moreno Vega, Departamento de Estadística, I.O. y Computación, Universidad de La Laguna, 38271 La Laguna. Nació el 31 de julio de 1967 en Gáldar, Las Palmas de Gran Canaria. Es Doctor en Informática y Profesor Titular de la Universidad de La Laguna. Ha sido miembro del comité de programa, comité organizador y revisor de varios congresos de Inteligencia Artificial y Heurísticas para la resolución de problemas. Ha liderado diversos proyectos de investigación dedicados al estudio de técnicas heurísticas en la resolución de problemas de optimización. Es coautor de artículos de investigación publicados en revistas internacionales como Studies in Locational Análisis, European Journal of Operational Research, Journal of Heuristics, Fuzzy Sets and Systems, BMC Bioinformatics y Parallel Computing, actuando como editor invitado en algunas de ellas.