Algoritmos Genéticos

Vania Paola Torrez Flores
vanis_59@hotmail.com

RESUMEN
Los algoritmos genéticos son métodos sistemáticos para la resolución de problemas de búsqueda y optimización que aplican a estos los mismos métodos de la evolución biológica: selección basada en la población, reproducción sexual y mutación.

Palabras Claves
Algoritmos genéticos, selección, mutación, crossover.

1. INTRODUCCIÓN
John Holland desde pequeño, se preguntaba cómo logra la naturaleza, crear seres cada vez más perfectos (aunque, como se ha visto, esto no es totalmente cierto, o en todo caso depende de qué entienda uno por perfecto). Lo curioso era que todo se lleva a cabo a base de interacciones locales entre individuos, y entre estos y lo que les rodea. No sabía la respuesta, pero tenía una cierta idea de como hallarla: tratando de hacer pequeños modelos de la naturaleza, que tuvieran alguna de sus características, y ver cómo funcionaban, para luego extrapolar sus conclusiones a la totalidad.

De hecho, ya de pequeño hacía simulaciones de batallas célebres con todos sus elementos: copiaba mapas y los cubría luego de pequeños ejércitos que se enfrentaban entre sí.

En los años 50 entró en contacto con los primeros ordenadores, donde pudo llevar a cabo algunas de sus ideas, aunque no se encontró con un ambiente intelectual fértil para propagarlas. Fue a principios de los 60, en la Universidad de Michigan en Ann Arbor, donde, dentro del grupo Logic of Computers, sus ideas comenzaron a desarrollarse y a dar frutos. Y fue, además, leyendo un libro escrito por un biólogo evolucionista, R. A. Fisher, titulado

La teoría genética de la selección natural, como comenzó a descubrir los medios de llevar a cabo sus propósitos de comprensión de la naturaleza. De ese libro aprendió que la evolución era una forma de adaptación más potente que el simple aprendizaje, y tomó la decisión de aplicar estas ideas para desarrollar programas bien adaptados para un fin determinado.

En esa universidad, Holland impartía un curso titulado Teoría de sistemas adaptativos, fue donde se crearon las ideas que más tarde se convertirían en los algoritmos genéticos.

Por tanto, cuando Holland se enfrentó a los algoritmos genéticos, los objetivos de su investigación fueron dos:

- imitar los procesos adaptativos de los sistemas naturales.
- diseñar sistemas artificiales (normalmente programas)

que retengan los mecanismos importantes de los sistemas naturales.

Unos 15 años más adelante, David Goldberg, actual delfín de los algoritmos genéticos, conoció a Holland, y se convirtió en su estudiante. Golberg era un ingeniero industrial trabajando en diseño de pipelines, y fue uno de los primeros que trató de aplicar los algoritmos genéticos a problemas industriales. Aunque Holland trató de disuadirle, porque pensaba que el problema era excesivamente complicado como para aplicarle algoritmos genéticos, Goldberg consiguió lo que quería, escribiendo un algoritmo genético en un ordenador personal Apple II. Estas y otras aplicaciones creadas por estudiantes de Holland convirtieron a los algoritmos genéticos en un campo con base suficiente aceptado para celebrar la primera conferencia en 1985, ICGA´85. Tal conferencia se sigue celebrando bianualmente.

John Koza: "Es un algoritmo matemático altamente paralelo que transforma un conjunto de objetos matemáticos individuales con respecto al tiempo usando operaciones modeladas de acuerdo al principio Darwiniano de reproducción y supervivencia del más apto, y tras haberse presentado de forma natural una serie de operaciones genéticas de entre las que destaca la recombinación sexual. Cada uno de estos objetos matemáticos suele ser una cadena de caracteres (letras o números) de longitud fija que se ajusta al modelo de las cadenas de cromosomas, y se les asocia con una cierta función matemática que refleja su aptitud".

2. MARCO TEÓRICO
La vida artificial es el estudio de la vida y de los sistemas artificiales que exhiben propiedades similares a los seres vivos, a través de modelos de simulación.

El área de vida artificial es un punto de encuentro para gente de otras áreas más tradicionales como lingüística, física, matemáticas, filosofía, psicología,Ciencias de la computación, biología, antropología y sociología en las que sería inusual que se discutieran enfoques teóricos y computacionales. Como área, tiene una historia controvertida; John Maynard Smith criticó ciertos trabajos de vida artificial en 1995 calificándolos de "ciencia sin hechos", y generalmente no ha recibido mucha atención de parte de biólogos. Sin embargo, la reciente publicación de artículos sobre vida artificial en revistas de amplia difusión, como Science y Nature son evidencia de que las técnicas de vida artificial son cada vez más aceptadas por los científicos, al menos como un método de estudio de la evolución.

2.1 ¿Qué es la vida Artificial?
La vida artificial pretende crear vida mediante la imitación de procesos y comportamientos de los seres humanos, todo esto con el objetivo de solucionar problemas del mundo real.

La vida artificial tiene como objetivo la aplicación del conocimiento sobre el funcionamiento del fenómeno denominado genéricamente "vida", dentro de la investigación y desarrollo de ciertas tecnologías. Esta es una nueva disciplina, en la cual se aplican principios biológicos a sistemas computacionales para la

resolución de problemas, principalmente sobre cooperación entre entidades individuales, como robots, en entornos cambiantes.

3. DESARROLLO

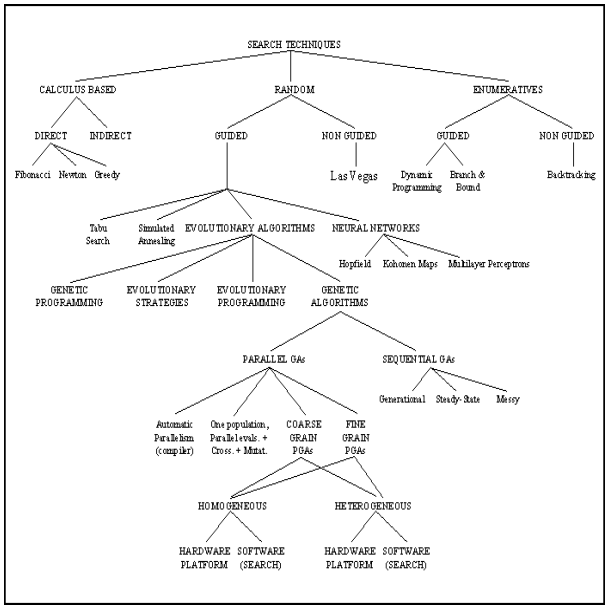

Figura 1. Cuadro Resumen

3.1 Anatomía De Un Algoritmo Genético Simple

```
BEGIN /* Algoritmo Genetico Simple */
    Generar una poblacion inicial.
    Computar la funcion de evaluacion de cada individuo.
    WHILE NOT Terminado DO
    BEGIN /* Producir nueva generacion */
        FOR Tamaño poblacion/2 DO
        BEGIN /*Ciclo Reproductivo */
            Seleccionar dos individuos de la anterior generacion,
            para el cruce (probabilidad de seleccion proporcional
            a la funcion de evaluacion del individuo).
            Cruzar con cierta probabilidad los dos
            individuos obteniendo dos descendientes.
            Mutar los dos descendientes con cierta probabilidad.
            Computar la funcion de evaluacion de los dos
            descendientes mutados.
            Insertar los dos descendientes mutados en la nueva generacion.
        END
        IF la poblacion ha convergido THEN
            Terminado := TRUE
    END
END
```

Figura 1: Pseudocódigo del Algoritmo Genético Simple

Figura 2.

Los algoritmos genéticos son métodos sistemáticos para la resolución de problemas de búsqueda y optimización que aplican a estos los mismos métodos de la evolución biológica: selección basada en la población, reproducción sexual y mutación.

Los algoritmos genéticos son métodos de optimización, que tratan de resolver el mismo conjunto de problemas que se ha contemplado anteriormente, es decir, hallar $(x_i,...,x_n)$ tales que $F(x_i,...,x_n)$ sea máximo. En un algoritmo genético, tras parametrizar el problema en una serie de variables, $(x_i,...,x_n)$ se codifican en un cromosoma.

Todas los operadores utilizados por un algoritmo genético se aplicarán sobre estos cromosomas, o sobre poblaciones de ellos. En el algoritmo genético va implícito el método para resolver el problema; son solo parámetros de tal método los que están codificados, a diferencia de otros algoritmos evolutivos como la programación genética. Hay que tener en cuenta que un algoritmo genético es independiente del problema, lo cual lo hace un algoritmo robusto, por ser útil para cualquier problema, pero a la vez débil, pues no está especializado en ninguno.

Las soluciones codificadas en un cromosoma compiten para ver cuál constituye la mejor solución (aunque no necesariamente la mejor de todas las soluciones posibles). El ambiente, constituido por las otras camaradas soluciones, ejercerá una presión selectiva sobre la población, de forma que sólo los mejor adaptados (aquellos que resuelvan mejor el problema) sobrevivan o leguen su material genético a las siguientes generaciones, igual que en la evolución de las especies. La diversidad genética se introduce mediante mutaciones y reproducción sexual.

En la Naturaleza lo único que hay que optimizar es la supervivencia, y eso significa a su vez maximizar diversos factores y minimizar otros. Un algoritmo genético, sin embargo, se usará habitualmente para optimizar sólo una función, no diversas funciones relacionadas entre sí simultáneamente. La optimización que busca diferentes objetivos simultáneamente, denominada multimodal o multiobjetivo, también se suele abordar con un algoritmo genético especializado.

Por lo tanto, un algoritmo genético consiste en lo siguiente: hallar de qué parámetros depende el problema, codificarlos en un cromosoma, y se aplican los métodos de la evolución: selección y reproducción sexual con intercambio de información y alteraciones que generan diversidad. En las siguientes secciones se verán cada uno de los aspectos de un algoritmo genético.

3.2 Funcionamiento

Los algoritmos genéticos establecen una analogía entre el conjunto de soluciones de un problema, llamado fenotipo, y el conjunto de individuos de una población natural, codificando la información de cada solución en una cadena, generalmente binaria, llamada cromosoma. Los símbolos que forman la cadena son llamados los genes. Cuando la representación de los cromosomas se hace con cadenas de dígitos binarios se le conoce como genotipo. Los cromosomas evolucionan a través de iteraciones, llamadas generaciones. En cada generación, los cromosomas son evaluados usando alguna medida de aptitud. Las siguientes generaciones (nuevos cromosomas), llamada descendencia, se forman utilizando dos operadores, de cruzamiento y de mutación.

3.3 Funcionamiento de un Algoritmo Genético Básico

Un algoritmo genético puede presentar diversas variaciones, dependiendo de cómo se aplican los operadores genéticos (cruzamiento, mutación), de cómo se realiza la selección y de cómo se decide el reemplazo de los individuos para formar la

nueva población. En general, el pseudocódigo consiste de los siguientes pasos:

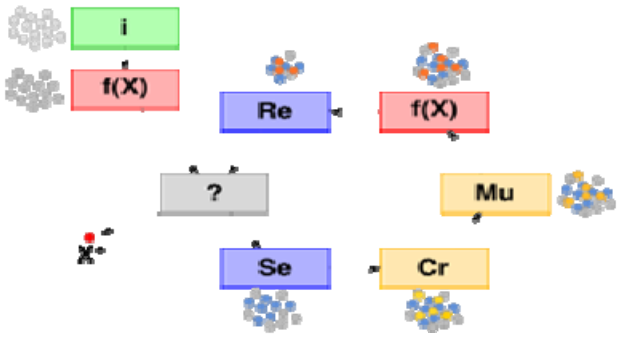

Figura 3. Algoritmo genético

3.3.1 Algoritmo Genético

Inicialización, f(X): evaluación?: condición de término, Se: selección, Cr: cruzamiento, Mu: mutación, Re: reemplazo, X*: mejor solución.

- Inicialización. Se genera aleatoriamente la población inicial, que está constituida por un conjunto de cromosomas los cuales representan las posibles soluciones del problema. En caso de no hacerlo aleatoriamente, es importante garantizar que dentro de la población inicial, se tenga la diversidad estructural de estas soluciones para tener una representación de la mayor parte de la población posible o al menos evitar la convergencia prematura.

- Evaluación. A cada uno de los cromosomas de esta población se aplicará la función de aptitud para saber qué tan "buena" es la solución que se está codificando.

- Condición de término. El AG se deberá detener cuando se alcance la solución óptima, pero ésta generalmente se desconoce, por lo que se deben utilizar otros criterios de detención. Normalmente se usan dos criterios: correr el AG un número máximo de iteraciones (generaciones) o detenerlo cuando no haya cambios en la población. Mientras no se cumpla la condición de término se hace lo siguiente:

  - Selección. Después de saber la aptitud de cada cromosoma se procede a elegir los cromosomas que serán cruzados en la siguiente generación. Los cromosomas con mejor aptitud tienen mayor probabilidad de ser seleccionados.

  - Cruzamiento. El cruzamiento es el principal operador genético, representa la reproducción sexual, opera sobre dos cromosomas a la vez para generar dos descendientes donde se combinan las características de ambos cromosomas padres.

  - Mutación. Modifica al azar parte del cromosoma de los individuos, y permite alcanzar zonas del espacio de búsqueda que no estaban cubiertas por los individuos de la población actual.

  - Reemplazo. una vez aplicados los operadores genéticos, se seleccionan los mejores individuos para conformar la población de la generación siguiente

3.3.2 Codificación de las Variables

Los algoritmos genéticos requieren que el conjunto se codifique en un cromosoma. Cada cromosoma tiene varios genes, que corresponden a sendos parámetros del problema. Para poder trabajar con estos genes en el ordenador, es necesario codificarlos en una cadena, es decir, una ristra de símbolos que generalmente va a estar compuesta de 0s y 1s.

Por ejemplo, en esta cadena de bits, el valor del parámetro p1 ocupará las posiciones 0 a 2, el p2 las 3 a 5, etcétera, como aparece en la tabla 2. El número de bits usado para cada parámetro dependerá de la precisión que se quiera en el mismo o del número de opciones posibles (alelos) que tenga ese parámetro. Por ejemplo, si se codifica una combinación del Mastermind, cada gen tendrá tantas opciones como colores halla, el número de bits elegido será el $\log_2$(número de colores).

Hay otras codificaciones posibles, usando alfabetos de diferente cardinalidad; sin embargo, uno de los resultados fundamentales en la teoría de algoritmos genéticos, el teorema de los esquemas, afirma que la codificación óptima, es decir, aquella sobre la que los algoritmos genéticos funcionan mejor, es aquella que tiene un alfabeto de cardinalidad 2.

Aquí se está codificando cada parámetro como un número entero de n bits. En realidad, se puede utilizar cualquier otra representación interna: bcd, código Gray y codificación en forma de números reales, por ejemplo.

La mayoría de las veces, una codificación correcta es la clave de una buena resolución del problema. Generalmente, la regla heurística que se utiliza es la llamada regla de los bloques de construcción, es decir, parámetros relacionados entre sí deben de estar cercanos en el cromosoma. Por ejemplo, si queremos codificar los pesos de una red neuronal, una buena elección será poner juntos todos los pesos que salgan de la misma neurona de la capa oculta (también llamada codificación fregona), como se indica en la figura. En esta, todos los pesos señalados con trazo doble se codifican mediante grupos de bits o bytes sucesivos en el cromosoma.

En todo caso, se puede ser bastante creativo con la codificación del problema, teniendo siempre en cuenta la regla anterior. Esto puede llevar a usar cromosomas bidimensionales, o tridimensionales, o con relaciones entre genes que no sean puramente lineales de vecindad. En algunos casos, cuando no se conoce de antemano el número de variables del problema, caben dos opciones: codificar también el número de variables, fijando un número máximo, o bien, lo cual es mucho más natural, crear un cromosoma que pueda variar de longitud. Para ello, claro está, se necesitan operadores genéticos que alteren la longitud.

Normalmente, la codificación es estática, pero en casos de optimización numérica, el número de bits dedicados a codificar un parámetro puede variar, o incluso lo que representen los bits

dedicados a codificar cada parámetro. Algunos paquetes de algoritmos genéticos adaptan automáticamente la codificación según van convergiendo los bits menos significativos de una solución.

3.3.3 Algoritmo Genético propiamente Dicho

Para comenzar la competición, se generan aleatoriamente una serie de cromosomas. El algoritmo genético procede de la forma siguiente:

Algoritmo genético:

1. Evaluar la puntuación (fitness) de cada uno de los genes.

2. Permitir a cada uno de los individuos reproducirse, de acuerdo con su puntuación.

3. Emparejar los individuos de la nueva población, haciendo que intercambien material genético, y que alguno de los bits de un gen se vea alterado debido a una mutación espontánea.

Cada uno de los pasos consiste en una actuación sobre las cadenas de bits, es decir, la aplicación de un operador a una cadena binaria. Se les denominan, por razones obvias, operadores genéticos, y hay tres principales: selección, crossover o recombinación y mutación; aparte de otros operadores genéticos no tan comunes, todos ellos se verán a continuación.

Un algoritmo genético tiene también una serie de parámetros que se tienen que fijar para cada ejecución, como los siguientes:

- Tamaño de la población: debe de ser suficiente para garantizar la diversidad de las soluciones, y, además, tiene que crecer más o menos con el número de bits del cromosoma, aunque nadie ha aclarado cómo tiene que hacerlo. Por supuesto, depende también del ordenador en el que se esté ejecutando.

- Condición de terminación: lo más habitual es que la condición de terminación sea la convergencia del algoritmo genético o un número prefijado de generaciones.

3.3.4 Operadores Genéticos

- Evaluación y selección. Durante la evaluación, se decodifica el gen, convirtiéndose en una serie de parámetros de un problema, se halla la solución del problema a partir de esos parámetros, y se le da una puntuación a esa solución en función de lo cerca que esté de la mejor solución. A esta puntuación se le llama fitness.

  El fitness determina siempre los cromosomas que se van a reproducir, y aquellos que se van a eliminar, pero hay varias formas de considerarlo para seleccionar la población de la siguiente generación:

  - Usar el orden, o rango, y hacer depender la probabilidad de permanencia o evaluación de la posición en el orden.

  - Aplicar una operación al fitness para escalarlo; como por ejemplo el escalado sigma.

  - En algunos casos, el fitness no es una sola cantidad, sino diversos números, que tienen diferente consideración. Basta con que tal fitness forme un orden parcial, es decir, que se puedan comparar dos individuos y decir cuál de ellos es mejor. Esto suele suceder cuando se necesitan optimizar varios objetivos.

  Una vez evaluado el fitness, se tiene que crear la nueva población teniendo en cuenta que los *buenos* rasgos de los mejores se transmitan a esta. Para ello, hay que seleccionar a una serie de individuos encargados de tan ardua tarea. Y esta selección, y la consiguiente reproducción, se puede hacer de dos formas principales:

  - Basado en el rango: en este esquema se mantiene un porcentaje de la población, generalmente la mayoría, para la siguiente generación. Se coloca toda la población por orden de fitness, y los M menos dignos son eliminados y sustituidos por la descendencia de alguno de los M mejores con algún otro individuo de la población. A este esquema se le pueden aplicar otros criterios; por ejemplo, se crea la descendencia de uno de los paladines/amazonas, y esta sustituye al más parecido entre los perdedores. Esto se denomina crowding, y fue introducido por DeJong. En nuestro caso, se eliminaría el cromosoma número 3, y se sustituiría por un descendiente del cromosoma número 2 y otro aleatorio, escogido entre el 1 y el 4. En realidad, para este esquema se escoge un crowding factor, CF. Cuando nace una nueva criatura, se seleccionan CF individuos de la población, y se elimina al más parecido a la nueva criatura.

    Una variante de este es el muestreado estocástico universal, que trata de evitar que los individuos con más fitness copen la población; en vez de dar la vuelta a una ruleta con una ranura, da la vuelta a la ruleta con N ranuras, tantas como la población; de esta forma, la distribución estadística de descendientes en la nueva población es más parecida a la real.

  - Rueda de ruleta: se crea un pool genético formado por cromosomas de la generación actual, en una cantidad proporcional a su fitness. Si la proporción hace que un individuo domine la población, se le aplica alguna operación de escalado. Dentro de este pool, se cogen parejas aleatorias de cromosomas y se emparejan, sin importar incluso que sean del mismo progenitor (para eso están otros operadores, como la mutación). Hay otras variantes: por ejemplo, en la nueva generación se puede incluir el mejor representante de la generación actual. En este caso, se denomina método elitista.

  - Selección de torneo: se escogen aleatoriamente un número T de individuos de la población, y el que tiene puntuación mayor se reproduce, sustituyendo su descendencia al que tiene menor puntuación.

- Crossover. Consiste en el intercambio de material genético entre dos cromosomas (a veces más, como el operador orgía propuesto por Eiben et al.). El crossover es el principal operador genético, hasta el punto que se puede decir que no es un algoritmo genético si no tiene crossover, y, sin

embargo, puede serlo perfectamente sin mutación, según descubrió Holland. El teorema de los esquemas confía en él para hallar la mejor solución a un problema, combinando soluciones parciales.

Para aplicar el crossover, entrecruzamiento o recombinación, se escogen aleatoriamente dos miembros de la población. No pasa nada si se emparejan dos descendientes de los mismos padres; ello garantiza la perpetuación de un individuo con buena puntuación (y, además, algo parecido ocurre en la realidad; es una práctica utilizada, por ejemplo, en la cría de ganado, llamada inbreeding, y destinada a potenciar ciertas características frente a otras). Sin embargo, si esto sucede demasiado a menudo, puede crear problemas: toda la población puede aparecer dominada por los descendientes de algún gen, que, además, puede tener caracteres no deseados. Esto se suele denominar en otros métodos de optimización atranque en un mínimo local, y es uno de los principales problemas con los que se enfrentan los que aplican algoritmos genéticos.

En cuanto al teorema de los esquemas, se basa en la noción de bloques de construcción. Una buena solución a un problema está constituido por unos buenos bloques, igual que una buena máquina está hecha por buenas piezas. El crossover es el encargado de mezclar bloques buenos que se encuentren en los diversos progenitores, y que serán los que den a los mismos una buena puntuación. La presión selectiva se encarga de que sólo los buenos bloques se perpetúen, y poco a poco vayan formando una buena solución. El teorema de los esquemas viene a decir que la cantidad de buenos bloques se va incrementando con el tiempo de ejecución de un algoritmo genético, y es el resultado teórico más importante en algoritmos genéticos.

El intercambio genético se puede llevar a cabo de muchas formas, pero hay dos grupos principales

- ***Crossover n-puntos*:** los dos cromosomas se cortan por n puntos, y el material genético situado entre ellos se intercambia. Lo más habitual es un crossover de un punto o de dos puntos.

- ***Crossover uniforme*:** se genera un patrón aleatorio de 1s y 0s, y se intercambian los bits de los dos cromosomas que coincidan donde hay un 1 en el patrón. O bien se genera un número aleatorio para cada bit, y si supera una determinada probabilidad se intercambia ese bit entre los dos cromosomas.

- Crossover especializados: en algunos problemas, aplicar aleatoriamente el crossover da lugar a cromosomas que codifican soluciones inválidas; en este caso hay que aplicar el crossover de forma que genere siempre soluciones válidas. Un ejemplo de estos son los operadores de crossover usados en el problema del viajante.

- Mutación: En la Evolución, una mutación es un suceso bastante poco común (sucede aproximadamente una de cada mil replicaciones), como ya se ha visto anteriormente. En la mayoría de los casos las mutaciones son letales, pero en promedio, contribuyen a la diversidad genética de la especie. En un algoritmo genético tendrán el mismo papel, y la misma frecuencia (es decir, muy baja).

Una vez establecida la frecuencia de mutación, por ejemplo, uno por mil, se examina cada bit de cada cadena cuando se vaya a crear la nueva criatura a partir de sus padres (normalmente se hace de forma simultánea al crossover). Si un número generado aleatoriamente está por debajo de esa probabilidad, se cambiará el bit (es decir, de 0 a 1 o de 1 a 0).

Si no, se dejará como está. Dependiendo del número de individuos que haya y del número de bits por individuo, puede resultar que las mutaciones sean extremadamente raras en una sola generación.

No hace falta decir que no conviene abusar de la mutación. Es cierto que es un mecanismo generador de diversidad, y, por tanto, la solución cuando un algoritmo genético está estancado, pero también es cierto que reduce el algoritmo genético a una búsqueda aleatoria. Siempre es más conveniente usar otros mecanismos de generación de diversidad, como aumentar el tamaño de la población, o garantizar la aleatoriedad de la población inicial.

Este operador, junto con la anterior y el método de selección de ruleta, constituyen un algoritmo genético simple, sga, introducido por Goldberg en su libro.

4. APLICACIONES

4.1 Mejorar Aptitud para la Siguiente Generación

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 cruzamiento, 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.

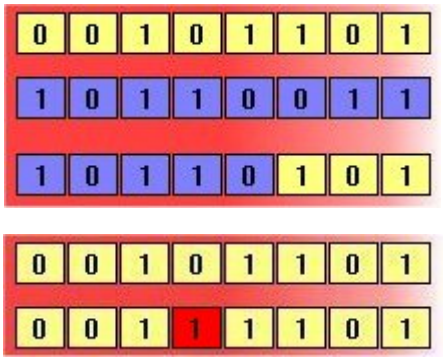

Figura 4. Cruzamiento y mutación.

El diagrama de arriba ilustra el efecto de estos dos operadores genéticos en los individuos de una población de cadenas de 8 bits.

El diagrama superior muestra a dos individuos llevando a cabo un cruzamiento de un punto; el punto de intercambio se establece entre las posiciones quinta y sexta del genoma, produciendo un nuevo individuo que es híbrido de sus progenitores. El segundo diagrama muestra a un individuo sufriendo una mutación en la posición 4, cambiando el 0 de esa posición de su genoma por un 1.

- ✓ Optimización de sistemas de compresión de datos, por ejemplo, usando wavelets.
- ✓ Predicción de Plegamiento de proteínas.
- ✓ Optimización de Layout.
- ✓ Predicción de estructura de RNA.
- ✓ En bio-informática, Alineamiento múltiple de secuencias.
- ✓ Aplicaciones en planificación de procesos industriales, incluyendo planificación job-shop.
- ✓ Selección optima de modelos matemáticos para la descripción de sistemas biológicos.
- ✓ Manejo de residuos sólidos.
- ✓ Ingeniería de software.
- ✓ Construcción de horarios en grandes universidades, evitando conflictos de clases.
- ✓ Problema del viajante.
- ✓ Hallazgo de errores en programas.
- ✓ Optimización de producción y distribución de energía eléctrica.

5. CONCLUSIÓN

En este sentido, se puede expresar que serían un modelo alternativo competitivo con los algoritmos genéticos, si se las programara para este fin. En rigor de verdades, la literatura sugiere que se podrían hacer modelos mixtos o híbridos en donde se combinen las ventajas de las redes neuronales y los algoritmos genéticos, aunque hay muy poco material disponible en este campo. Tal vez esto se deba al hecho que los GA y el estudio de las redes forman dos ramas o escuelas separadas dentro de la inteligencia artificial, por lo que existe una preferencia en los investigadores en perfeccionar alguno de los dos modelos antes que tratar de unirlos.

6. BIBLIOGRAFÍA

[1] http://www.geocities.com/RecearchTriangle/Lab/5196/redesn.html

[2] http://electronica.com.mx/neural/articulos/in

[3] http://www.une.edu.ve/electronica/neurona.htm

[4] http//geneura.ugr.es/~jmerelo/ie/ags.htm

[5] http://www.aernsoft.com/cas/ayuda/apendic12.htm

[6] http://www.iamnet.com/users/jcontre/genetic