ALGORITMOS GENÉTICOS

Arranz de la Peña, Jorge
Universidad Carlos III
100025106@alumnos.uc3m.es

Parra Truyol, Antonio
Universidad Carlos III
100023822@alumnos.uc3m.es

En este documento se pretende analizar los algoritmos genéticos descubriendo su funcionamiento y sus secretos.

1. INTRODUCCION

Cuando hablamos de algoritmos genéticos, hay que hablar de John Holland que en 1962 asienta las bases para sus posteriores desarrollos hasta llegar a lo que se conoce hoy por algoritmos genéticos.

Un algoritmo genético es un método de búsqueda que imita la teoría de la evolución biológica de Darwin para la resolución de problemas. Para ello, se parte de una población inicial de la cual se seleccionan los individuos más capacitados para luego reproducirlos y mutarlos para finalmente obtener la siguiente generación de individuos que estarán más adaptados que la anterior generación.

2. ESQUEMA BÁSICO

En la naturaleza todo el proceso de evolución biológica se hace de forma natural pero para aplicar el algoritmo genético al campo de la resolución de problemas habrá que seguir una serie de pasos. Una premisa es conseguir que el tamaño de la población sea lo suficientemente grande para garantizar la diversidad de soluciones. Se aconseja que la población sea generada de forma aleatoria para obtener dicha diversidad. En caso de que la población no sea generada de forma aleatoria habrá que tener en cuenta que se garantice una cierta diversidad en la población generada. Los pasos básicos de un algoritmo genético son:

- Evaluar la puntuación de cada uno de los cromosomas generados.
- Permitir la reproducción de los cromosomas siendo los más aptos los que tengan más probabilidad de reproducirse.
- Con cierta probabilidad de mutación, mutar un gen del nuevo individuo generado.
- Organizar la nueva población.

Estos pasos se repetirán hasta que se de una condición de terminación. Se puede fijar un número máximo de iteraciones antes de finalizar el algoritmo genético o detenerlo cuando no se produzcan más cambios en la población (convergencia del algoritmo). Esta última opción suele ser la más habitual.

Veamos el esquema general de un algoritmo genético simple:

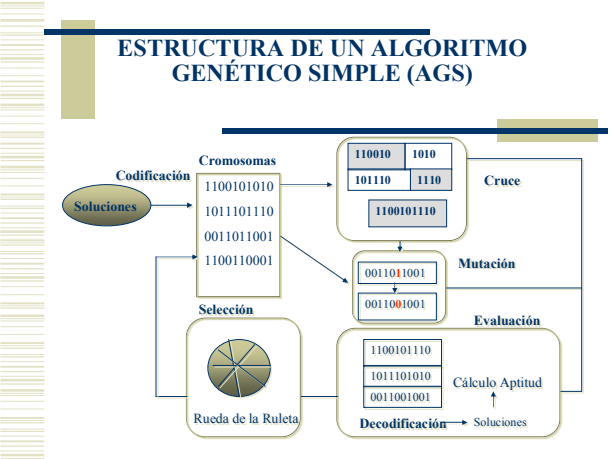

Figura1. Esquema de un algoritmo genético simple.

3. PARÁMETROS DE LOS ALGORITMOS GENÉTICOS.

Para el estudio de los algoritmos genéticos hay que tener en cuenta una serie de parámetros:

3.1 Tamaño de la Población

Este parámetro nos indica el número de cromosomas que tenemos en nuestra población para una generación determinada. En caso de que esta medida sea insuficiente, el algoritmo genético tiene pocas posibilidades de realizar reproducciones con lo que se realizaría una búsqueda de soluciones escasa y poco óptima. Por otro lado si la población es excesiva, el algoritmo genético será excesivamente lento. De hecho estudios revelan que hay un límite a partir del cual es ineficiente elevar el tamaño de la población puesto que no se consigue una mayor velocidad en la resolución del problema.

3.2 Probabilidad de Cruce

Indica la frecuencia con la que se producen cruces entre los cromosomas padre es decir, que haya probabilidad de reproducción entre ellos. En caso de que no exista probabilidad de reproducción, los hijos serán copias exactas se los padres. En caso de haberla, los hijos tendrán partes de los cromosomas de los padres. Si la probabilidad de cruce es del 100% el hijo se crea totalmente por cruce, no por partes.

3.3 Probabilidad de Mutación

Nos indica la frecuencia con la que los genes de un cromosoma son mutados. Si no hay mutación, los descendientes son los mismos que había tras la reproducción. En caso de que haya mutaciones, parte del cromosoma descendiente es modificado y si la probabilidad de mutación es del 100%, la totalidad del cromosoma se cambia. En este caso, no se cambian simplemente unos bits del cromosoma sino que se cambian todos, lo que

significa que se produce una inversión en el cromosoma y no una mutación por lo que la población degenera muy rápidamente.

4. OPERACIONES DE LOS ALGORITMOS GENÉTICOS.

Tras parametrizar el problema en una serie de variables, se codifican en un cromosoma. Todos 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. Hay que tener en cuenta que un algoritmo genético es independiente del problema, lo cual lo hace un algoritmo robusto, al resultar útil en cualquier ámbito de acción, 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 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.

Por lo tanto, un algoritmo genético consiste en hallar de qué parámetros depende el problema, codificarlos en un cromosoma, y aplicar los métodos de la evolución: selección y reproducción sexual con intercambio de información y mutaciones que generen diversidad.

4.1 Codificación de las Variables

Los cromosomas de alguna manera deberán contener información acerca de la solución que representa. La codificación se puede realizar de varias formas. La más utilizada es mediante una cadena de números binarios (1s o 0s). Pero también se puede realizar la codificación mediante números enteros o incluso cadenas de palabras.

La elección de la codificación dependerá también del problema a resolver pues puede darse la situación en la que la resolución de un caso sea más óptimo el uso de una codificación basada en números reales mientras que esa codificación complique la solución en otro caso. Así pues hay que estudiar la codificación más óptima según el caso que se esté estudiando.

4.1.1 Codificación Binaria

Es la codificación más extendida debido a que los primeros algoritmos genéticos utilizaron este tipo de codificación. En este caso, cada cromosoma es una cadena de bits (0 o 1). A su favor tiene que puede abarcar muchos cromosomas incluso con un número reducido de genes. Sin embargo por otro lado esta opción no es la idónea para muchos problemas y en algunas ocasiones es necesario realizar correcciones después de la reproducción y/o mutación.

Este tipo de codificación se utiliza por ejemplo en el problema de la mochila. En este problema tenemos una mochila con una cierta capacidad y una serie de objetos que queremos introducir. Estos objetos tendrán un peso y un beneficio o un valor para nosotros. La capacidad de la mochila es inferior a la suma del peso de todos los objetos. El objetivo es conseguir que la suma de los beneficios o valores sea máxima y que la suma de los pesos no supere el de la capacidad de la mochila.

4.1.2 Codificación Numérica

En este tipo de codificación se utilizan cadenas de números que representan un número en una secuencia. Se utiliza en problemas en los que hay que ordenar algo, donde resulta muy útil. En algunos casos también es necesario como en el caso anterior realizar correcciones tras relaciones o mutaciones.

Un ejemplo clásico de este tipo de codificación es el problema del viajante de comercio. En este caso tenemos una serie de ciudades por las que el comerciante debe pasar y las distancias entre ellas. El objetivo es que el comerciante, partiendo de una ciudad origen, recorra todas las ciudades y vuelva al punto de partida pero recorriendo el menor número de kilómetros, por lo que habrá que localizar la combinación de ciudades que minimice dicho recorrido.

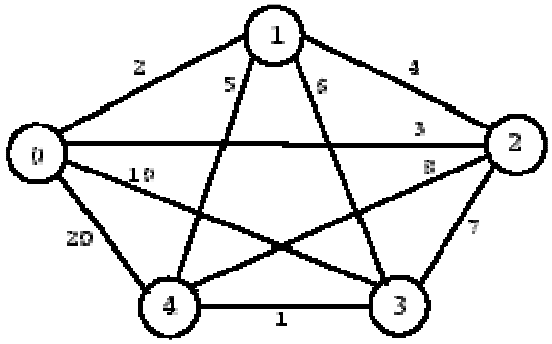

Figura 2. Esquema del problema del viajante

4.1.3 Codificación por Valor Directo

Este tipo de codificación será el utilizado en caso de resolución de problemas en el que se requiera del uso de valores de cifrado complicado como podría ser en el uso de números reales, cuya codificación con números binarios sería muy complejo. En codificación por valor directo cada cromosoma es una cadena de valores relacionados con el problema a estudiar, pudiendo ser desde números decimales, cadenas de caracteres o incluso una combinación de varios de ellos. Su aplicación es muy buena en ciertos problemas concretos. Por el contrario para la utilización de esta codificación, normalmente es necesario desarrollar nuevas técnicas de reproducción y mutación especificas hacia la resolución del problema.

Una aplicación de esta codificación se da en la resolución de problemas para la búsqueda de pesos para las redes neuronales. En este asunto se trata de encontrar el peso de las neuronas para ciertas entradas y así entrenar a la red para obtener la salida deseada. El valor del peso de las entradas vendrá representado en el propio cromosoma con dicha codificación.

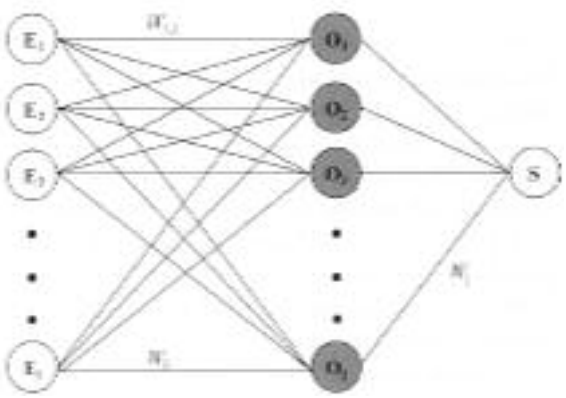

Figura 3. Esquema red neuronal

4.1.4 Codificación en Árbol

Este tipo de codificación se utiliza principalmente en el desarrollo de programas o expresiones para programación genética. Cada cromosoma será en este caso un árbol con ciertos objetos.

En este método, los cambios aleatorios pueden generarse cambiando el operador, alterando el valor de un cierto nodo del árbol o simplemente sustituyendo un subárbol por otro.

4.2 Selección

Como ya hemos visto anteriormente es necesario hacer una selección con los individuos más capacitados para que éstos sean los que se reproduzcan con más probabilidad de acuerdo con la teoría de Darwin en la cual los más capacitados son los que deben sobrevivir y crear una nueva descendencia más facultada.

Por lo tanto una vez evaluado cada cromosoma y obtenida su puntuación, se tiene que crear la nueva población teniendo en cuenta que los buenos rasgos de los mejores se transmitan a ésta.

Esta selección se puede realizar de varias formas como se verá a continuación.

4.2.1 Selección por Rueda de Ruleta

Se crea para esta selección una ruleta con los cromosomas presentes en una generación. Cada cromosoma tendrá una parte de esa ruleta mayor o menor en función a la puntuación que tenga cada uno. Se hace girar la ruleta y se selección el cromosoma en el que se para la ruleta. Obviamente el cromosoma con mayor puntuación saldrá con mayor probabilidad. En caso de que las probabilidades difieran mucho, este método de selección dará problemas puesto que si un cromosoma tiene un 90% de posibilidades de ser seleccionado, el resto apenas saldrá lo que reduciría la diversidad genética.

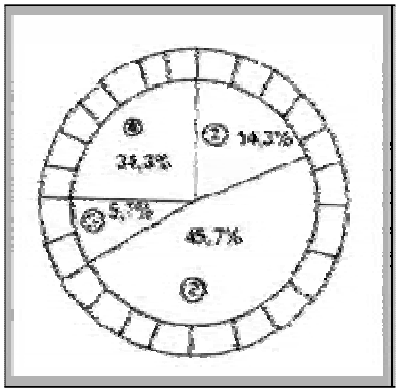

Figura 4. Rueda de ruleta de selección.

4.2.2 Selección por Rango

En este método a cada cromosoma se le asigna un rango numérico basado en su aptitud y la selección se realiza en base a este ranking. Veamos la diferencia entre el caso anterior y este a través de un ejemplo gráfico.

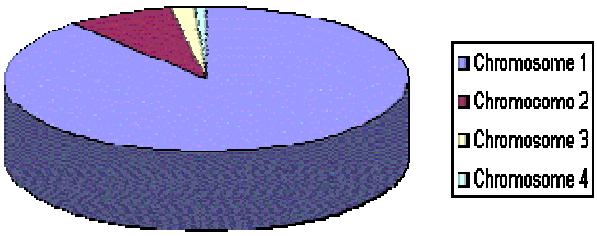

Figura 5. Caso de selección por ruleta.

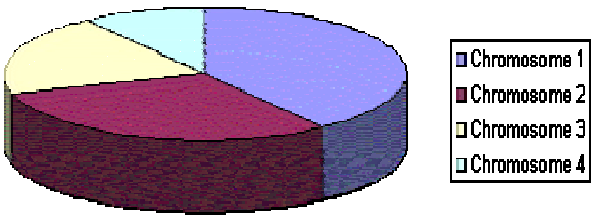

Figura 6. Caso de selección por ranking.

Vemos como en este último caso se va a producir una variedad genética mucho más rica que en el primer caso. El problema de esta selección es que la convergencia puede ser más lenta ya que no existe tanta diferencia entre el mejor cromosoma y el resto como ocurría antes.

4.2.3 Selección Elitista

En ciertas ocasiones puede suceder que tras el cruce y la mutación, perdamos el cromosoma con mejor adaptación. Este método de selección copia el mejor cromosoma o alguno de los mejores en la nueva población. El resto se realiza de la misma forma que hemos visto anteriormente. El elitismo puede mejorar el funcionamiento de los algoritmos genéticos al evitar que se

pierda la mejor solución. Una variación del elitismo es que el mejor cromosoma solo se copie a la siguiente generación en caso de que tras una reproducción/mutación no se haya generado un cromosoma mejor.

4.2.4 Selección por Estado Estacionario

La descendencia de los individuos seleccionados en cada generación vuelve a la población genética preexistente, reemplazando a algunos de los miembros menos aptos de la anterior generación. Se conservan algunos individuos entre generaciones.

4.2.5 Selección por Torneo

Se escogen de forma aleatoria un número de individuos de la población, y el que tiene puntuación mayor se reproduce, sustituyendo su descendencia al que tiene menor puntuación.

4.2.6 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.

4.2.7 Selección Jerárquica

En esta selección, 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.

4.2.8 Otras Selecciones

Existen otras técnicas de selección que simplemente se comentarán a continuación.

Una de ellas es la selección por prueba de aptitud en las que los cromosomas con más aptitud tienen más posibilidad de ser seleccionados pero no la certeza.

La selección generacional en la que ningún miembro de la población anterior se encuentra en la nueva.

4.3 Reproducción o Crossover

Una vez se realiza la selección de los cromosomas se procede a realizar la reproducción o cruce entre dos de estos cromosomas

Más concretamente, el crossover consiste en el intercambio de material genético entre dos cromosomas. El objetivo del cruce es conseguir que el descendiente mejore la aptitud de sus padres.

Para aplicar el cruce habrá que seleccionar con anterioridad dos individuos de la población con una de las diversas técnicas de selección que hemos mencionado en el punto anterior. Además esta selección puede elegir el mismo padre para un descendiente. Esto no es ningún problema pues se asegura la perpetuación del cromosoma más dominante pero si este cruce se produjese con mucha frecuiencia podría acarrear consecuencias adversas en caso de que ese cromosoma dominante presente algunos genes no deseados. Hay diferentes formas de realizar los cruces para cada codificación siendo algunas técnicas aplicables indistintamente a las distintas codificaciones como veremos a continuación.

4.3.1 Crossover 1 Punto

Los dos cromosomas padres se cortan por un punto. Se copia la información genética de uno de los padres desde el inicio hasta el punto de cruce y el resto se copia del otro progenitor. Es una de las formas clásicas de crossover.

Veamos algunos ejemplos gráficos:

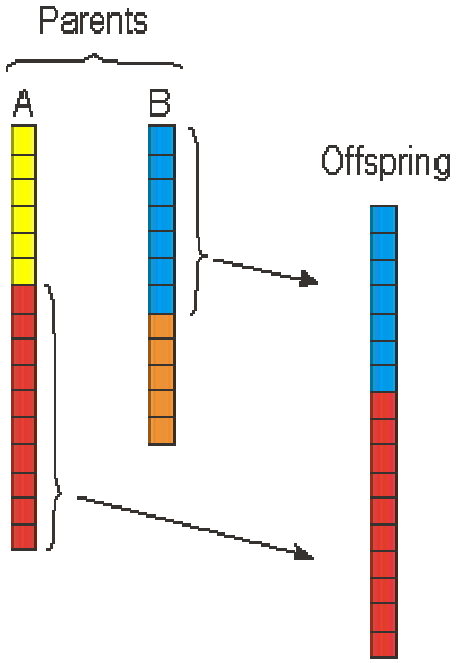

Figura 7. Crossover 1 punto para codificación binaria.

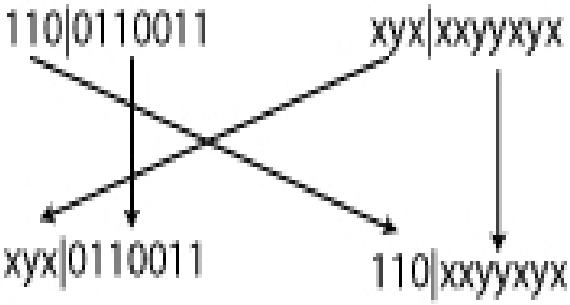

Figura 8. Crossover punto para codificación por valor directo.

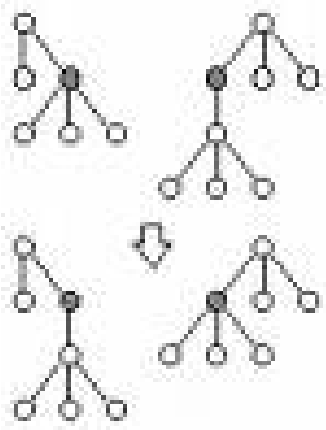

Figura 9. Crossover punto para codificación en árbol.

4.3.2 Crossover 2 Puntos

Se trata de la misma filosofía que en el caso anterior peo en este caso los padres se cortan por dos puntos. Se copiará al descendiente los genes de un cromosoma progenitor desde el principio hasta el primer punto de cruce, los genes del otro progenitor desde el primer punto de cruce hasta el segundo y del segundo punto de cruce hasta el final se copiará del otro progenitor.

Viendo unos ejemplos gráficos:

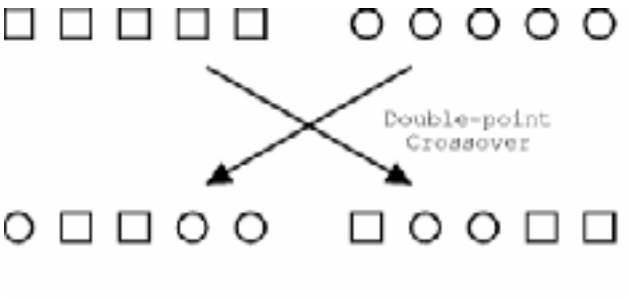

Figura 10. Crossover 2 puntos para codificación por valor directo

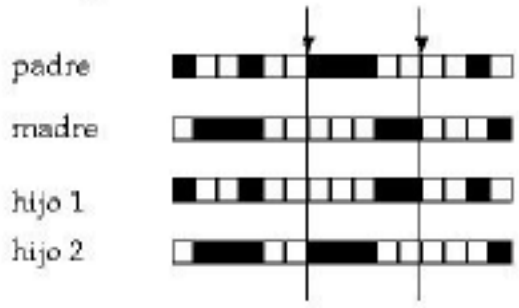

Figura 11. Crossover 2 puntos para codificación binaria

4.3.3 Crossover Uniforme

Cada gen del descendiente se obtiene de cualquiera de los padres de forma aleatoria. Una opción es generar un número aleatorio. Si este número supera un cierto umbral se elegirá un padre determinado y si no lo supera se elige al otro.

Veamos algún ejemplo:

Figura 12. Crossover uniforme para codificación binaria.

Parent 1	MIA	SFO	YEG	YHZ	YVR	YWG	YUL	YUL	YYZ
Parent 2	SFO	YEG	YHZ	YWG	YWG	YUL	YYZ	YYZ	YYZ
Random Bit String	0	1	1	1	0	1	0	0	0
Offspring 1	SFO	SFO	YEG	YHZ	YWG	YWG	YYZ	YYZ	YYZ
Offspring 2	MIA	YEG	YHZ	YWG	YVR	YUL	YUL	YUL	YYZ

Figura 13. Crossover uniforme para codificación por valor directo.

Otra opción es seleccionar una máscara. En caso de que el bit correspondiente a la máscara esté a 1, se copia el gen de un progenitor y en caso de que esté a 0 se copia el gen del otro progenitor.

Veamos un ejemplo:

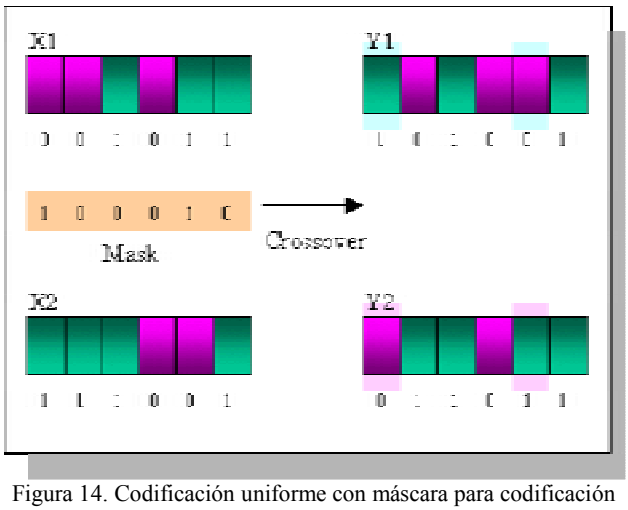

Figura 14. Codificación uniforme con máscara para codificación binaria.

4.3.4 Crossover Aritmético

Los progenitores se recombinan según algún operador aritmético para generar su descendiente.

Veamos algunos ejemplos:

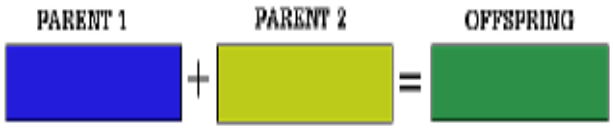

Figura 15. Crossover aritmético genérico.

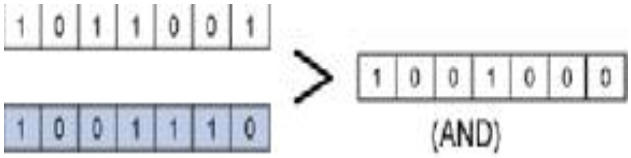

Figura 16. Crossover realizado con un operador AND.

4.4 Mutación

Tras el cruce, tiene lugar la mutación. Si nos referimos en términos de evolución, la mutación se manifiesta de forma extraordinaria, nada común. Las mutaciones suelen en promedio ser beneficiosas pues contribuyen a la diversidad genética de la especie. Además previenen a las soluciones de la población de verse limitadas por un óptimo local. Por lo tanto la mutación consiste en modificar ciertos genes de forma aleatoria atendiendo a la probabilidad de mutación establecida con anterioridad. La mutación depende de la codificación y de la reproducción. Si se abusa de la mutación podemos caer en el uso del algoritmo genético como una simple búsqueda aleatoria. Por lo tanto antes de aumentar las mutaciones, conviene estudiar otras soluciones que aporten diversidad a la población como podría ser el aumento del tamaño de la población o garantizar la aleatoriedad de la población inicial.

Para el caso de una codificación binaria, la mutación consiste simplemente en la inversión del gen mutado que corresponderá con un bit. En el caso de una codificación numérica, la mutación podría consistir en sustituir un número por otro o intercambiar un número por otro que está en otra posición del cromosoma. En el caso de codificación por valor directo en el que por ejemplo usemos números reales, la mutación puede consistir simplemente en modificar el valor en unos decimales. Por último, en una codificación en árbol, la mutación podría radicar en el cambio de operador, de un número o incluso en la mutación de una rama entera.

Veamos unos ejemplos para analizar el fenómeno de la mutación:

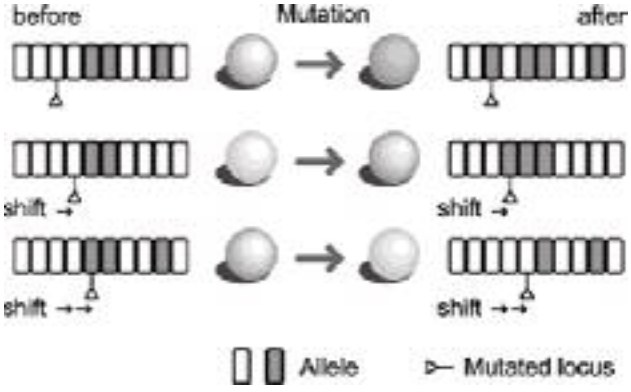

Figura 17. Mutación simple para una codificación binaria.

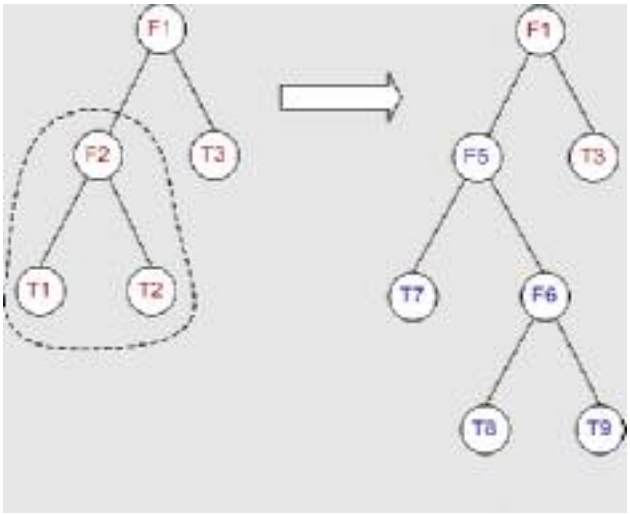

Figura 18. Mutación de rama para una codificación en árbol.

5. OTROS OPERADORES

En algunos problemas se pueden utilizar otro tipo de operadores que buscan soluciones de forma más ordenada o que actúan en las últimas fases para optimizar la solución.

5.1 Cromosomas de longitud variable.

Puede suceder que para ciertas aplicaciones como en las redes neuronales no se conozca de antemano el número de neuronas que vamos a utilizar por lo que habrá que disponer de alguna técnica para solventar el problema.

En estos casos, necesitamos dos operadores más: *añadir* y *eliminar*. Estos operadores se utilizan para añadir un gen, o eliminar un gen del cromosoma. La forma más habitual es *duplicar* uno ya existente, el cual sufre mutación y se añade al lado del anterior. En este caso, los operadores del algoritmo genético simple (selección, mutación, crossover) funcionarán de la forma habitual, salvo, claro está, que sólo se hará crossover en la zona del cromosoma de menor longitud. Estos operadores permiten, además, crear un *algoritmo genético de dos niveles*: a nivel de cromosoma y a nivel de gen.

5.2 Operadores de Nicho

Estos operadores están encaminados a mantener la diversidad genética de la población, de forma que cromosomas similares sustituyan sólo a cromosomas similares, y son especialmente útiles en problemas con muchas soluciones. Un algoritmo genético con estos operadores es capaz de hallar todos los máximos, dedicándose cada especie a un máximo. Más que operadores genéticos, son formas de enfocar la selección y la evaluación de la población.

6. VENTAJAS DE LOS ALGORITMOS GENÉTICOS

- Una clara ventaja es que los algoritmos genéticos son intrínsicamente paralelos, es decir, operan de forma simultánea con varias soluciones, en vez de trabajar de forma secuencial como las técnicas tradicionales. Esto significa que mientras técnicas tradicionales 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, los algoritmos genéticos simplemente desechan esta solución subóptima y siguen por otros caminos.
- Cuando se usan para problemas de optimización resultan menos afectados por los máximos locales (falsas soluciones) que las técnicas tradicionales. Muchos algoritmos de búsqueda pueden quedar atrapados en los óptimos locales: si llegan a lo alto de una colina del paisaje adaptativo, descubrirán que no existen soluciones mejores en las cercanías y concluirán que han alcanzado la mejor de todas, aunque existan picos más altos en algún otro lugar del mapa, situación que no sucede para algoritmos genéticos.
- Otra ventaja es su habilidad para manipular muchos parámetros simultáneamente. Resulta interesante en caso de tener varios objetivos a resolver.
- No necesitan conocimientos específicos sobre el problema que intentan resolver. Realizan cambios aleatorios en sus soluciones candidatas y luego utilizan la función de aptitud para determinar si esos cambios producen una mejora o no.
- Resulta sumamente fácil ejecutarlos en las modernas arquitecturas masivas en paralelo.
- Usan operadores probabilísticos, en vez de los típicos operadores determinísticos de las otras técnicas.

7. DESVENTAJAS DE LOS ALGORITMOS GENETICOS

- Definir una representación del problema. El lenguaje utilizado para especificar soluciones candidatas debe ser robusto, debe ser capaz de tolerar cambios aleatorios que no produzcan constantemente errores fatales o resultados sin sentido. Se puede solucionar mediante la definición de los individuos como listas de números donde cada número representa algún aspecto de la solución candidata.
- Pueden tardar mucho en converger, o no converger en absoluto, dependiendo en cierta medida de los parámetros que se utilicen -tamaño de la población, número de generaciones…
- Pueden converger prematuramente debido a una serie de problemas. 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.

8. APLICACIONES DE LOS ALGORITMOS GENÉTICOS

La aplicación más común de los algoritmos genéticos ha sido la solución de problemas de optimización, en donde han mostrado ser muy eficientes. Sin embargo, no todos los problemas pudieran ser apropiados para esta técnica. Se recomienda en general tomar en cuenta las siguientes características del mismo antes de intentar usarla:

- Su espacio de búsqueda debe estar delimitado dentro de un cierto rango.
- Debe poderse definir una función de aptitud que nos indique qué tan buena o mala es una cierta respuesta.
- Las soluciones deben codificarse de una forma que resulte relativamente fácil de implementar en la computadora.

Dentro de los distintos problemas de optimización podemos encontrar unas áreas de aplicación:

- Diseño por computadora de nuevos materiales que cumplan múltiples objetivos.
- Optimización del la carga de containers.
- Asignación de procesos en topologías de redes con procesamiento distribuido.
- Ubicación de archivos en sistemas de almacenamiento distribuido.
- Diseño de circuitos integrados.
- Optimización de la infraestructura de telefonía celular.
- Ingeniería Aeroespacial.
- Juegos.
- Robótica

9. EJEMPLOS PRÁCTICOS

En el siguiente enlace, se puede apreciar de forma práctica los conocimientos expuestos en relación a los algoritmos genéticos.

http://homepage.sunrise.ch/homepage/pglaus/gentore.htm#Applet

En esta página se encuentra un applet de Java en el que se muestra un relieve de un determinado paisaje generado de forma aleatoria. Se puede elegir la población inicial. Al ejecutar el programa, el algoritmo trata de buscar la máxima altura a la que podemos ascender.

Otro ejemplo de algoritmo genético se puede encontrar en http://www.rennard.org/alife/english/gavgb.html donde el algoritmo trata de buscar una copia de una figura partiendo de otras generadas de forma aleatoria.

10. CONCLUSIONES

Como hemos podido ver a lo largo del documento los algoritmos genéticos son actualmente una fuerte fuente de resolución de problemas complejos al realizar su ejecución en paralelo pudiendo así obtener diferentes soluciones al problema.

Sin embargo, como se ha podido observar, no hay ninguna estrategia que sea siempre invencible, sino que hay un conjunto de estrategias que suelen dar buenos resultados. Así pues, habrá que ajustar los parámetros de acción en función de cada problema a modelar para obtener una solución que se adapte mejor a unas determinadas condiciones. Sin embargo, en una situación real puede suceder que no se conozcan los parámetros iniciales o que no se sepa la duración del algoritmo.

Por lo tanto habrá que elegir con sumo cuidad los parámetros iniciales. Un crossover elevado es recomendable para la mayoría de los problemas aunque hay situaciones en las que un crossover menor dará mejores resultados. La mutación deberá ser baja, entorno al 1%. Para una óptima resolución, el resto de parámetros habrá que determinarlos en función del problema. La población a elegir debe atender a un valor óptimo (no por tener una población mayor, se va a tener una solución mejor), la codificación dependerá del problema a resolver como se ha visto con anterioridad y por último el método de selección más utilizado es el de selección por rueda de ruleta, aunque como se ha comentado, dependerá de la cuestión que nos trate.

Por último, se pretende que estos algoritmos se parezcan lo más posible a lo que en realidad sucede en la naturaleza. Por ejemplo, cuando una población queda aislada en la naturaleza, como sucede, por ejemplo, con los linces de Doñana, se pierde diversidad al haber un grupo reducido que se reproduce entre si constantemente perdiéndose esa tan necesaria diversidad para sobrevivir. De hecho se ha comprobado que no evolucionan y que su adaptación a las condiciones no mejora en función del cambio de éstas.

Por lo tanto hay que tener cuidado con la perdida de diversidad en nuestra población, que se puede conseguir entre otras formas mediante la mutación.

11. REFERENCIAS

[1] http://es.wikipedia.org/wiki/Algoritmo_gen%C3%A9tico

[2] http://www.redcientifica.com/doc/doc199904260011.html

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

[4] http://www.lsi.upc.es/~iea/transpas/9_geneticos/index.htm

[5] http://casa.ccp.servidores.net/genetico.html

[6] http://homepage.sunrise.ch/homepage/pglaus/gentore.htm#Applet

[7] http://the-geek.org/docs/algen/

[8] http://www.alfredorahn.com/docs/AG_Clase_3.ppt

[9] http://www.alfredorahn.com/docs/AG_Clase_4.ppt

[10] http://www.fiec.espol.edu.ec/investigacion/topico/algoritmos geneticos.pdf

[11] http://www.rennard.org/alife/english/gavgb.html