Cátedra de: ALGORITMOS GENÉTICOS

"INTRODUCCIÓN A LOS ALGORITMOS GENÉTICOS" (continuación)

Profesora: Ing. Daniela Díaz

Universidad Tecnológica Nacional
Facultad Regional Rosario

Año: 2013

Población

Tamaño de la población

Una cuestión que se suele plantear es la relacionada con el tamaño idóneo de la población. Parece intuitivo que las poblaciones pequeñas corren el riesgo de no cubrir adecuadamente el espacio de búsqueda, mientras que el trabajar con poblaciones de gran tamaño puede acarrear problemas relacionados con el excesivo costo computacional.

Goldberg[1] efectuó un estudio teórico, obteniendo como conclusión que el tamaño óptimo de la población para ristras[2] de longitud l, con codificación binaria, crece exponencialmente con el tamaño de la ristra.

Este resultado traería como consecuencia que la aplicabilidad de los Algoritmos Genéticos en problemas reales sería muy limitada, ya que resultarían no competitivos con otros métodos de optimización combinatoria. Basándose en evidencia empírica se sugiere que un tamaño de población comprendida entre 1 y 21 es suficiente para atacar con éxito los problemas por el considerados.

Población inicial

Habitualmente la población inicial se escoge generando ristras al azar, pudiendo contener cada gen uno de los posibles valores del alfabeto con probabilidad uniforme. Nos podríamos preguntar que es lo que sucedería si los individuos de la población inicial se obtuviesen como resultado de alguna técnica heurística o de optimización local. En los pocos trabajos que existen sobre este aspecto, se constata que esta inicialización no aleatoria de la población inicial, puede acelerar la convergencia del Algoritmo Genético. Sin embargo en algunos casos la desventaja resulta ser la prematura convergencia del algoritmo, queriendo indicar con esto la convergencia hacia óptimos locales.

Función objetivo

Dos aspectos que resultan cruciales en el comportamiento de los Algoritmos Genéticos son la determinación de una adecuada función de adaptación o función objetivo, así como la codificación utilizada.

Idealmente nos interesaría construir funciones objetivo con "ciertas regularidades", es decir funciones objetivo que verifiquen que para dos individuos que se encuentren cercanos en el espacio de búsqueda, sus respectivos valores en las funciones objetivo sean similares. Por otra parte una dificultad en el comportamiento del Algoritmo Genético puede ser la existencia de gran cantidad de óptimos locales, así como el hecho de que el óptimo global se encuentre muy aislado.

La regla, general para construir una buena función objetivo es que ésta debe reflejar el valor del individuo de una manera "real", pero en muchos problemas de optimización combinatoria, donde existe gran cantidad de restricciones, buena parte de los puntos del espacio de búsqueda representan individuos no válidos.

Para este planteamiento en el que los individuos están sometidos a restricciones, se han propuesto varias soluciones. La primera sería la que podríamos denominar absolutista, en la que aquellos individuos que no verifican las restricciones, no son considerados como tales, y se siguen efectuando cruces y mutaciones hasta obtener individuos válidos, o bien, a dichos individuos se les asigna una función objetivo igual a cero.

[1] Golber: ingeniero industrial, trabajó en diseño de *pipelines*, y fue uno de los primeros que trató de aplicar los algoritmos genéticos a problemas industriales.

[2] Ristra: cadena. Ristra de símbolos (números o letras) que generalmente va a estar compuesta de 0s y 1s.

Otra posibilidad consiste en reconstruir aquellos individuos que no verifican las restricciones. Dicha reconstrucción suele llevarse a cabo por medio de un nuevo operador que se acostumbra a denominar reparador.

Otro enfoque está basado en la penalización de la función objetivo. La idea general consiste en dividir la función objetivo del individuo por una cantidad (la penalización) que guarda relación con las restricciones que dicho individuo viola. Dicha cantidad puede simplemente tener en cuenta el número de restricciones violadas o bien el denominado costo esperado de reconstrucción, es decir el coste asociado a la conversión de dicho individuo en otro que no viole ninguna restricción.

Otra técnica que se ha venido utilizando en el caso en que la computación de la función objetivo sea muy compleja es la denominada evaluación aproximada de la función objetivo. En algunos casos la obtención de n funciones objetivo aproximadas puede resultar mejor que la evaluación exacta de una única función objetivo (supuesto el caso de que la evaluación aproximada resulta como mínimo n veces más rápida que la, evaluación exacta).

Un problema habitual en las ejecuciones de los Algoritmos Genéticos surge debido a la velocidad con la que el algoritmo converge. En algunos casos la convergencia es muy rápida, lo que suele denominarse *convergencia prematura*, en la cual el algoritmo converge hacia óptimos locales, mientras que en otros casos el problema es justo el contrario, es decir se produce una convergencia lenta del algoritmo. Una posible solución a estos problemas pasa por efectuar transformaciones en la función objetivo.

El problema de la convergencia prematura, surge a menudo cuando la selección de individuos se realiza de manera proporcional a su función objetivo. En tal caso, pueden existir individuos con una adaptación al problema muy superior al resto, que a medida que avanza el algoritmo "dominan" a la población. Por medio de una transformación de la función objetivo, en este caso una *comprensión* del rango de variación de la función objetivo, se pretende que dichos "superindividuos" no lleguen a dominar a la población.

El problema de la lenta convergencia del algoritmo, se resolvería de manera análoga, pero en este caso efectuando una *expansión* del rango de la función objetivo.

La idea de especies de organismos, ha sido imitada en el diseño de los Algoritmos Genéticos en un método propuesto por Goldberg y Richardson, utilizando una modificación de la función objetivo de cada individuo, de tal manera que individuos que estén muy cercanos entre sí devalúen su función objetivo, con objeto de que la población gane en diversidad.

Selección

La función de selección de padres más utilizada, es la denominada función de selección proporcional a la función objetivo, en la cual cada individuo tiene una, probabilidad de ser seleccionado como padre que es proporcional al valor de su función objetivo.

Denotando por (p super prop sub j,t) la probabilidad de que el individuo (I super j sub t) sea seleccionado como padre, se tiene que:

$$p_{j,t}^{\text{prop}} = \frac{g(I_t^j)}{\sum_{j=1}^{\lambda} g(I_t^j)}.$$

Esta función de selección es invariante ante un cambio de escala, pero no ante una traslación.

Una de las maneras de superar el problema relacionado con la rápida convergencia proveniente de los superindividuos, que surge al aplicar la anterior función de selección, es el efectuar la selección proporcional al rango del individuo, con lo cual se produce una repartición más uniforme de la probabilidad de selección. Si denotamos por rango(g(I super j sub t)) el rango de la función objetivo del individuo (I super j sub t) cuando los individuos de la población han sido ordenados de menor a mayor (es decir el peor individuo tiene rango 1, mientras que el individuo con mejor función objetivo tiene rango lambda), y sea (p super rango sub j,t) la probabilidad de que el individuo (I super j sub t) sea seleccionado como padre cuando la selección se efectúa proporcionalmente al rango del individuo, se tiene que

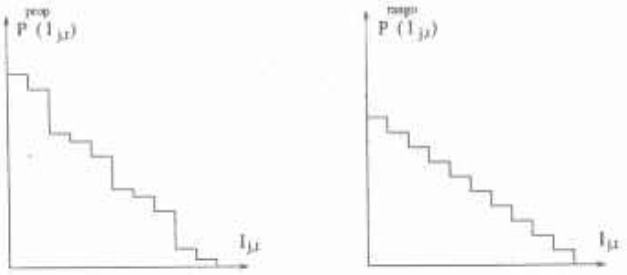


Figura 6: Esquemas de selección de padres proporcional a la función objetivo (izquierda) y proporcional al rango de la función objetivo (derecha)

$$p_{j,t}^{\text{rango}} = \frac{\text{rango}(g(I_t^j))}{\lambda(\lambda+1)/2}.$$

La suma de los rangos, lambda(lambda + 1)/2, constituye la constante de normalización.

La función de selección basada en el rango es invariante frente a la translación y al cambio de escala.

Otro posible refinamiento del modelo de selección proporcional, es el modelo de selección del valor esperado, el cual actúa de la manera siguiente: para, cada individuo If, se introduce un contador, inicialiazado en g(I super j sub t)/gt, donde, gt denota la media, de.) a función objetivo en la generación t. Cada vez que el individuo (I super j sub t) es seleccionado para el cruce, dicho contador decrece en una cantidad c (c pertenece a (0, 5;.1)). El individuo en cuestión dejará de poder ser seleccionado en esa generación, cuando su contador sea negativo.

Un esquema de selección, introducido por Brindle, y que empíricamente ha proporcionado buenos resultados, es el denominado muestreo estocástico con reemplazamiento del resto, en el cual cada individuo es seleccionado un número de veces que coincide con la parte entera del número esperado de ocurrencias de dicho suceso, compitiendo los individuos por los restos. Es decir, si denotamos por n(I super j sub t) el número de veces que el individuo (I super j sub t) es seleccionado para el cruce, tenemos que:

$$n(I_t^j) = n_1(I_t^j) + n_2(I_t^j), \text{ donde}$$

$n_1(I_t^j) = [g(I_t^j)/\bar{g}_t]$, y $n_2(I_t^j)$ corresponde a la componente aleatoria que resulta de muestrear sobre los $\lambda - \sum_{j=1}^{\lambda}[g(I_t^j)/\bar{g}_t]$ restantes individuos, siendo el muestro proporcional a la función objetivo de cada individuo.

Baker introduce un método denominado muestreo universal estocástico, el cual utiliza un único giro de la ruleta siendo los sectores circulares proporcionales a la función objetivo. Los individuos son seleccionados a partir de marcadores, igualmente espaciados y con comienzo aleatorio.

Efectuando un paralelismo con los métodos de muestreo estadísticos, este último tipo de selección de padres se relaciona con el muestreo sistemático, mientras que la selección proporcional a la función objetivo, está basada en el muestreo estratificado con fijación proporcional al tamaño.

También el procedimiento de selección que hemos denominado muestreo estocástico con reemplazamiento del resto, mantiene un paralelismo con el muestreo estratificado con fijación de compromiso.

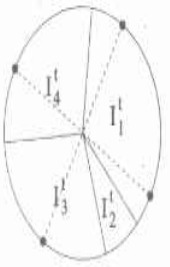


Figura 7: Método de selección de padres denominado muestreo universal estocástico. El individuo I_1^t se escoge 2 veces, mientras que I_3^t e I_4^t son elegidos una única vez

En el modelo de selección elitista se fuerza a que el mejor individuo de la población en el tiempo t, sea seleccionado como padre.

La selección por torneo, constituye un procedimiento de selección de padres muy extendido y en el cual la idea consiste en escoger al azar un número de individuos de la población, tamaño del torneo, (con o sin reemplazamiento), seleccionar el mejor individuo de este grupo, y repetir el proceso hasta que el número de individuos seleccionados coincida con el tamaño de la población. Habitualmente el tamaño del torneo es 2, y en tal caso se ha utilizado una versión probabilística en la cual se permite la selección de individuos sin que necesariamente sean los mejores.

Una posible clasificación de procedimientos de selección de padres consistirá en: métodos de selección dinámicos, en los cuales las probabilidades de selección varían de generación a generación, (por ejemplo la selección proporcional a la función objetivo), frente a métodos de selección estáticos, en los cuales dichas probabilidades permanecen constantes (por ejemplo la selección basada en rangos).

Si se asegura que todos los individuos tienen asignada una probabilidad de selección distinta de cero el método de selección se denomina preservativo. En caso contrario se acostumbra a denominarlo extintivo.

Cruce

El Algoritmo Genético Canónico descrito en el apunte anterior, utiliza el cruce basado en un punto, en el cual los dos individuos seleccionados para jugar el papel de padres, son recombinados por medio de la selección de un punto de corte, para posteriormente intercambiar las secciones que se encuentran a la derecha de dicho punto.

Se han investigado otros operadores de cruce, habitualmente teniendo en cuenta más de un punto de cruce. Se investigó el comportamiento del operador de cruce basado en múltiples puntos, concluyendo que el cruce basado en dos puntos, representaba una mejora mientras que añadir más puntos de cruce no beneficiaba el comportamiento del algoritmo. La ventaja de tener más de un punto de cruce radica en que el espacio de búsqueda puede ser explorado más fácilmente, siendo la principal desventaja el hecho de aumentar la probabilidad de ruptura de buenos esquemas.

En el operador de cruce basado en dos puntos, los cromosomas (individuos) pueden contemplarse como un circuito en el cual se efectúa la selección aleatoria de dos puntos, tal y como se indica en la figura.

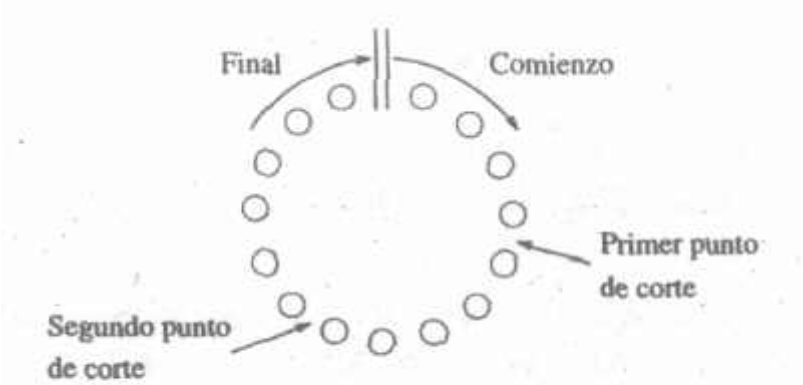


Figura 8: Individuo visto como un circuito

Desde este punto de vista, el cruce basado en un punto, puede verse como un caso particular del cruce basado en dos puntos, en el cual uno de los puntos de corte se encuentra fijo al comienzo de la ristra que representa al individuo.

En el denominado operador de cruce uniforme (Syswerda) cada gen, en la descendencia se crea copiando el correspondiente gen de uno de los dos padres, escogido de acuerdo a una "máscara de cruce" generada aleatoriamente. Cuando existe un l en la "máscara de cruce", el gen es copiado del primer padre, mientras que cuando exista un 0 en la "máscara de cruce", el gen se copia del segundo padre.

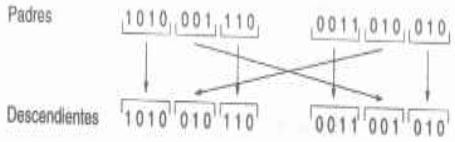


Figura 9: Operador de cruce basado en dos puntos

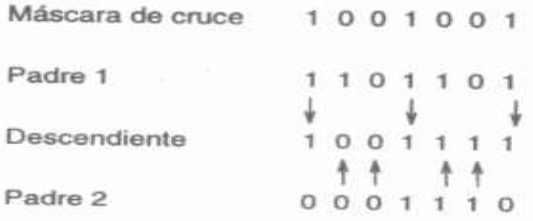


Figura 10: Operador de cruce uniforme

En la literatura, el término operador de cruce uniforme se relaciona con la obtención de la "máscara de cruce" uniforme, en el sentido de que cualquiera de los elementos del alfabeto tenga asociada la misma probabilidad. Hablando en términos de la teoría de la probabilidad la máscara de cruce está compuesta por una muestra aleatoria de tamaño A extraída de una distribución de probabilidad de Bernouilli de parámetro 1/2.

Si tuviésemos en cuenta el valor de la función de adaptación de cada padre en el momento de generar la "máscara de cruce", de tal manera que cuanto mayor sea la función de adaptación de un individuo, más probable sea heredar sus características, podríamos definir, un operador de cruce basado en la función objetivo, en el cual la "máscara de cruce" se interpreta como una muestra aleatoria de tamaño 1 proveniente de una distribución de Bernouilli de parámetro donde (I super j sub t) y I(super i sub t) denotan los padres seleccionados para ser cruzados.

$$p = g(I_t^j)/(g(I_t^j) + g(I_t^i))$$

El concepto de "máscara de cruce" puede también servir para representar los cruces basados en un punto y basados en múltiples puntos, tal y como se muestra en .

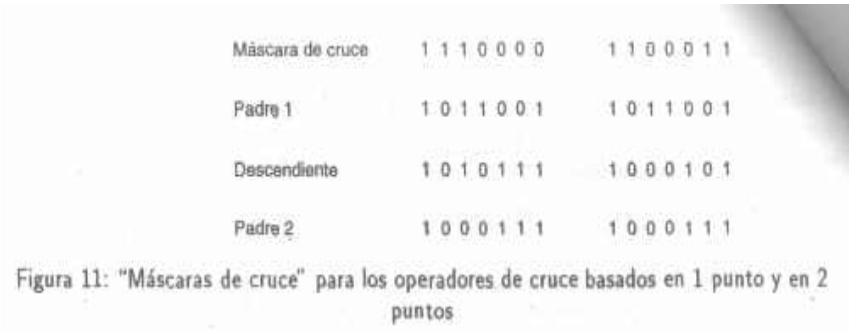


Figura 11: "Máscaras de cruce" para los operadores de cruce basados en 1 punto y en 2 puntos

Existen otros operadores de cruce específicos para un determinado problema como son, por ejemplo, los definidos para el problema del agente de comercio (problema del viajante).

Podemos explicar el crossover cíclico a través del siguiente ejemplo:

Supongamos que tomamos 2 cromosomas de una determinada longitud, para el ejemplo podría ser de 10 genes.

padre 1

9	8	2	1	7	4	5	10	6	3

padre 2

1	2	3	4	5	6	7	8	9	10

Para el caso del problema del viajante estos dos cromosomas corresponden también a recorridos dos de los recorridos.

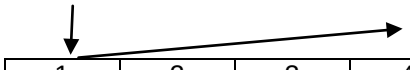
El primer hijo se obtiene tomando el primer gen del primer padre 1, es decir el 9**.**

9	8	2	1	7	4	5	10	6	3

1	2	3	4	5	6	7	8	9	10

Se observa la configuración de los padres, y se ve que debajo del 9 del primer padre hay un 1 en el segundo padre.

9	8	2	1	7	4	5	10	6	3



1	2	3	4	5	6	7	8	9	10

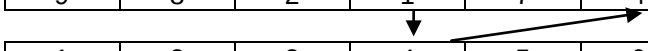
Luego se busca el 1 en el primer padre, el cual corresponde al cuarto gen y se toma este gen.

Entonces ya tenemos los genes:

9			1						

Debajo del 1 del primer padre notamos que en la configuración se presenta un 4.

9	8	2	1	7	4	5	10	6	3



1	2	3	4	5	6	7	8	9	10

Elegimos en el primer padre el 4 en el lugar correspondiente y ya tenemos:

9			1		4				

En la configuración inicial notamos debajo del 4 hay un 6 que lo buscamos en el primer padre.

9	8	2	1	7	4	5	10	6	3
1	2	3	4	5	6	7	8	9	10

Luego obtenemos:

9			1		4			6	

Observamos debajo del último gen seleccionado (el 6) y aparece el 9 que ya lo habíamos utilizado.
Entonces termina el ciclo y aquellos genes que no tienen valor se completan con los genees correspondientes al segundo padre.
Luego obtenemos un primer hijo con los genes:

9	2	3	1	5	4	7	8	6	10

En forma simétrica se obtiene el segundo hijo:

1	8	2	4	7	6	5	10	9	3

Mutación

La mutación se considera un operador básico, que proporciona un pequeño elemento de aleatoriedad en la vecindad (entorno) de los individuos de la población. Si bien se admite que el operador de cruce es el responsable de efectuar la búsqueda a lo largo del espacio de posibles soluciones, también parece desprenderse de los experimentos efectuados por varios investigadores que el operador de mutación va ganando en importancia a medida que la población de individuos va convergiendo.

La búsqueda del valor óptimo para la probabilidad de mutación, es una cuestión que ha sido motivo de varios trabajos.

Si bien en la mayoría de las implementaciones de Algoritmos Genéticos se asume que tanto la probabilidad de cruce como la de mutación permanecen constantes, algunos autores han obtenido mejores resultados experimentales modificando la probabilidad de mutación a medida que aumenta el número de iteraciones.

El operador de Mutación se aplica inmediatamente después del de Crossover, normalmente a un porcentaje bajo de la población. • En general es considerado un operador secundario, de menor importancia que Selección y Crossover, dado que se aplica a un bajo porcentaje de la población y su efecto no es demasiado notable en la mayoría de los casos.

Sin embargo existen Algoritmos Evolutivos (no Genéticos) cuyo principal operador de búsqueda es la mutación. La idea detrás de los operadores de Mutación, es reproducir las mutaciones genéticas producidas en cada generación (provenientes de errores de copia o transferencia en el ADN, etc.), como una de las principales herramientas de la Evolución Natural. En los AG, el operador de Mutación tiene varios objetivos:

- preservar la diversidad genética de la población evitando la convergencia prematura,
- explorar áreas posiblemente no abordadas del espacio de búsqueda (y cercanas a una buena solución),
- sacar al AG de un máximo local si se produjo convergencia prematura.

La Mutación se utiliza en bajo porcentaje (entre el 1 y el 5 % en codificación binaria o finita), debido al peligro de que opere sobre la única copia disponible de una buena solución y la arruine.
En general esto no sucede, ya que las buenas soluciones reciben varias copias y es poco probable que se muten todas. Sin embargo, existen casos especiales como los Niching[3] Genetic Algorithms donde hay muchas posibilidades que esto ocurra, por lo que en general no utilizan mutación.

Un porcentaje excesivo de Mutación provoca que la búsqueda se convierta en aleatoria (dado que gran cantidad de soluciones son mutadas al azar en cada generación); un porcentaje demasiado bajo puede provocar convergencia prematura, o que ciertas zonas del espacio de búsqueda no sean exploradas. Sin embargo, este efecto es en general menos notable que el de la Selección o Crossover.

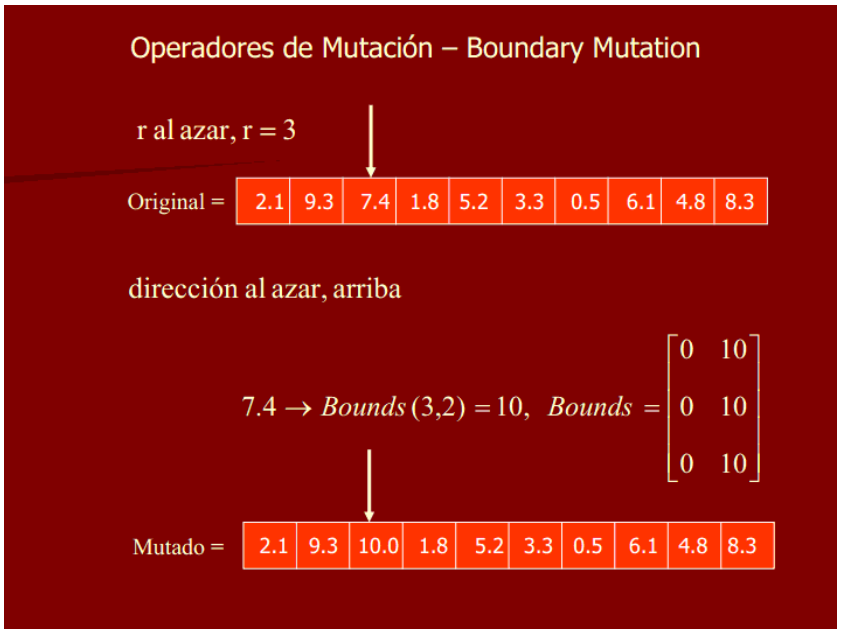
[3] Los Métodos de Niching son mecanismos que pueden resolver problemas teniendo la capacidad de crear y mantener varias subpoblaciones dentro de un determinado espacio de búsqueda. Cada una de estas subpoblaciones corresponde a cada óptimo que se pretende encontrar de una determinada función multimodal.

Operadores de Mutación

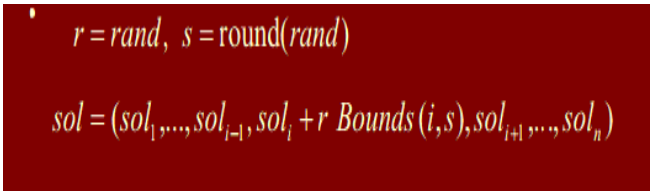
Boundary Mutation (Mutación al Borde)
Se aplica en general a codificaciones Reales, Enteras, o Finitas(no Binarias ni basadas en el orden).
Cambia alguna de las coordenadas (elegida al azar) por su valor en un extremo del intervalo de definición de la variable (dada por la matriz de Bounds)

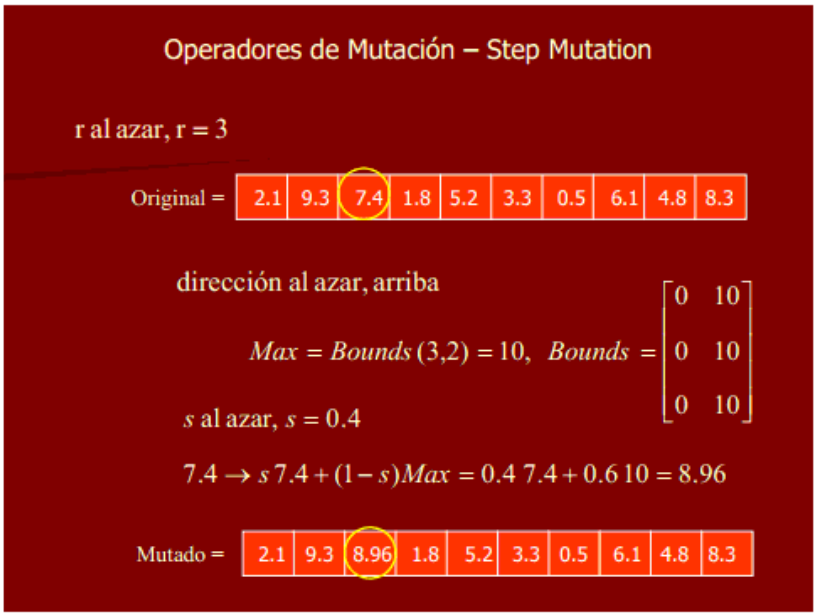
Se elige una coordenada al azar del individuo seleccionado, se elige una dirección de cambio (hacia el máximo o hacia el mínimo) y se cambia el valor de la coordenada seleccionada por el valor máximo o mínimo posible, según lo elegido.

Se aplica principalmente con la intención de corregir la dificultad que tienen los AG en general, para encontrar soluciones ubicadas en los bordes

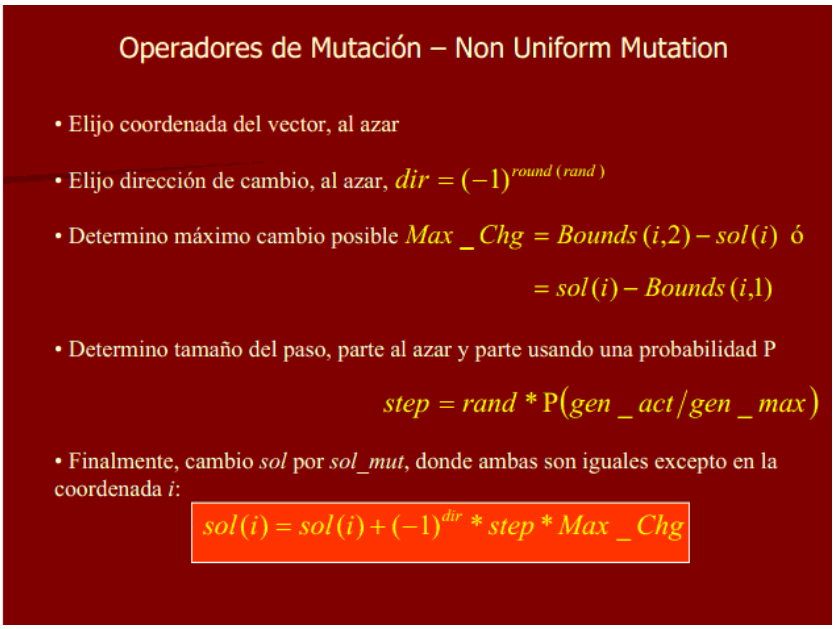


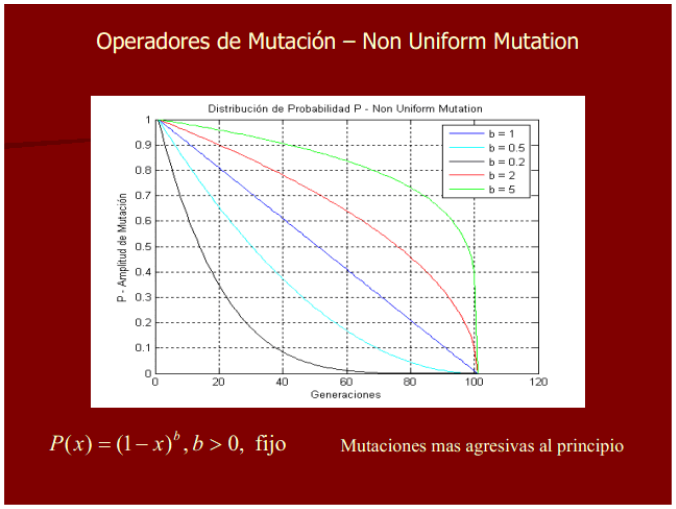
Step Mutation (También llamada Uniform Mutation dado que la probabilidad de mutación usada es uniforme)
Similar a Boundary Mutation
Se aplica en general a codificaciones Reales, Enteras, o Finitas(no Binarias ni basadas en el orden).
Cambia alguna de las coordenadas (elegida al azar) por un valor intermedio entre el actual y un extremo del intervalo de definición de la variable (dada por la matriz de Bounds)
Se elige una coordenada al azar del individuo seleccionado, se elige una dirección de cambio (hacia el máximo o hacia el mínimo) y se elige un valor al azar (entre 0 y 1)





Non Uniform Mutation •
Similar a Step Mutation, pero incluye una Probabilidad de Mutación que va bajando con el número de Generaciones.
Se aplica en general a codificaciones Reales, Enteras, o Finitas(no Binarias ni basadas en el orden)
Al inicio del proceso hay mucha mutación, al avanzar el proceso, el sistema se va "congelando" y permite cada vez menos permutaciones.
Para esto utiliza como parámetros la generación actual, el número total de generaciones previsto, y una distribución de probabilidad P, que al principio genera grandes mutaciones (entre 0 y uno de los bordes del dominio Bounds, al azar).
Conforme aumenta el número de Generaciones, el sistema se va congelando, y P genera mutaciones cada vez menores, hasta que eventualmente P llega a 0 y se congela, no generando más mutaciones al alcanzar el número estipulado de generaciones.





NonUniform Mutation Modificada

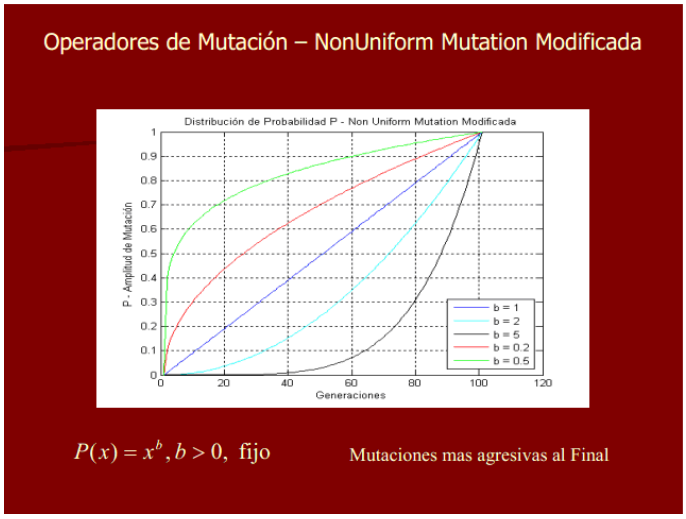
Similar a Non Uniform Mutation

Se aplica en general a codificaciones Reales, Enteras, o Finitas (no Binarias ni basadas en el orden).

En Algoritmos Genéticos, la mutación sirve entre otras cosas para evitar óptimos locales, o sea, salir de crowding cuando se produce prematuramente.

En ese caso, una mayor proporción de mutaciones, y de mayor intensidad y extensión, son útiles cerca del fin de la corrida, y no al principio.

Cambiamos entonces en el procedimiento anterior, la probabilidad P de modo que empiece en 0 y termine en 1.



Multi NonUniform Mutation

Similar a Non Uniform Mutation pero muta todas las coordenadas del vector.

Se aplica en general a codificaciones Reales, Enteras, o Finitas (no Binarias ni basadas en el orden).

Se determina vector de direcciones de cambio, al azar *dir* = (−1)*round* (*rand* (1,*n*))

Se determina vector de máximos cambios posibles

Max _ Chg (*i*) = *Bounds* (*i*,2) − *sol* (*i*) ó *sol* (*i*) − *Bounds* (*i*,1)

Se determina el cambio en cada coordenada, de forma similar a Non Uniform Mutation ó Modified Non Uniform Mutation

step = *rand*(1,*n*).**P*(*gen_ act gen_max*)

Y Finalmente,

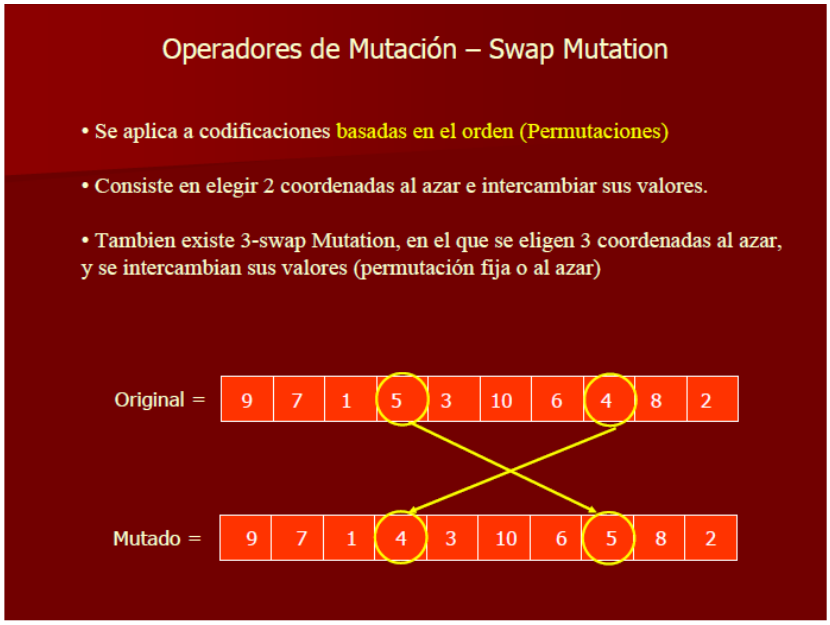
sol = sol + dir . step .*Max _ Chg* (Cambia TODAS las coordenadas)

NonUniform Mutations
Uniform Mutation => cambia 1 coordenada a la vez. El cambio es al azar en el intervalo seleccionado (todo el posible dado por Bounds), y no cambia a lo largo de la corrida

Non Uniform Mutation => cambia 1 coordenada a la vez. El cambio es al azar, dentro del intervalo seleccionado, pero este intervalo comienza siendo todo el posible dado por Bounds, y termina siendo un punto (el sistema se va "Congelando")

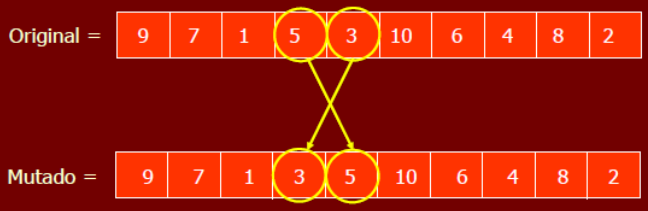
Multi Non Uniform Mutation => cambia TODAS las coordenada a la vez. El cambio es al azar, dentro del intervalo seleccionado, pero este intervalo comienza siendo todo el posible dado por Bounds, y termina siendo un punto (el sistema se va "Congelando")

Modified NonUniform and Multi Non Uniform Mutations: => Idem, pero los intervalos aumentan en vez de disminuir, con la intención de sacar al algoritmo de los posibles óptimos locales que encuentre.



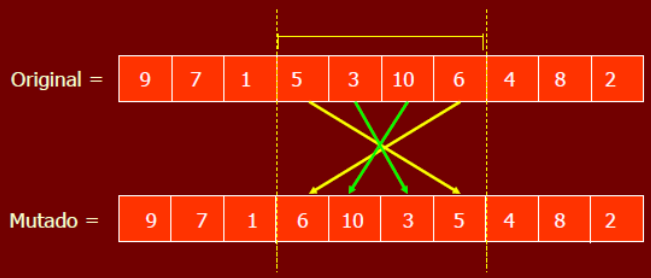
Operadores de Mutación – AdjointSwap Mutation

- Se aplica a codificaciones basadas en el orden (Permutaciones)
- Modificación de Swap Mutation
- Consiste en elegir una coordenada al azar, e intercambiarla con la siguiente



Operadores de Mutación – Inversion Mutation

- Se aplica a codificaciones basadas en el orden (Permutaciones)
- Generalización de Adjoint Swap Mutation
- Se eligen 2 coordenadas al azar, y se Invierte el orden de recorrido entre los puntos elegidos



Binary Mutation

Se aplica a codificaciones Binarias.
Se crea una máscara binaria al azar (similar a Uniform Crossover).
Las coordenadas con 1s son cambiadas de valor, de 1 a 0 o de 0 a 1 según corresponda.
Hay una versión probabilística, donde se fija un Umbral de probabilidades, se genera un vector al azar (coordenadas reales entre 0 y 1), y se cambian sólo aquellas probabilidades que tienen valores mayores que el umbral prefijado.

Cuando los valores de cada elemento de la máscara supera la probabilidad (en el ejemplo P= 0,4), se cambian los ceros por los unos y viceversa.

Arquitectura de un Algoritmo Genético

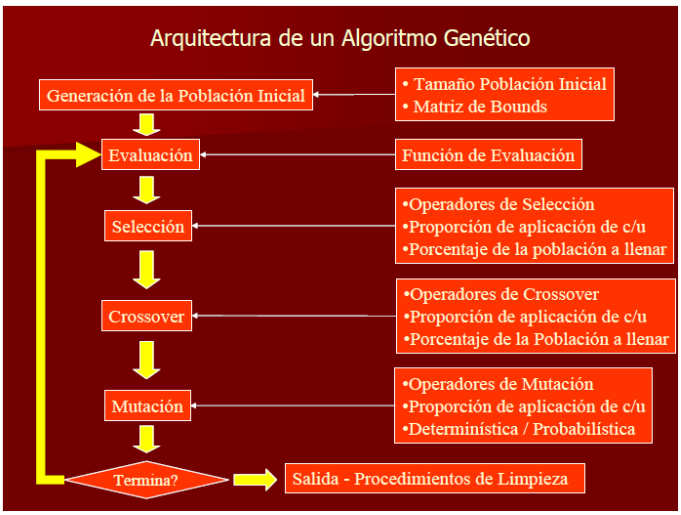
La Arquitectura de un Algoritmo Genético consiste en todas las decisiones estructurales que permiten su aplicación en un caso particular.
Una vez decidido que el Algoritmo será generacional (existen otros tipos), basado en el esquema común de Selección, Cruzamiento y Mutación, queda un número de decisiones a tomar:

- Tamaño de la Población Inicial
- Cantidad Total de Generaciones
- Se aplicará Elitismo? (preservar parte de la población intacta)
- Proporción de la población a la que se aplicará Crossover
- Cómo se llenará el resto de la población, que no sea producida por mutación o Crossover?
- La población será de tamaño constante, o puede variar durante la corrida?
- • etc.

Algunas de estas decisiones (como cantidad de generaciones) son fácilmente modificables entre corridas, pero otras (como el Elitismo), necesitan ajustes de código para ser modificadas.

Características a definir:

- Codificación del Problema: la cual puede ser Binaria, Finita, Real, etc.
- Función de Evaluación
- Matriz de Bounds
- Tipo de AG: por ejemplo Generacional
- Operadores de Selección: Tipos y proporción de aplicación
- Operadores de Crossover: Tipos y proporción de aplicación
- Operadores de Mutación: Tipos y proporción de aplicación
- Tamaño de la Población Inicial
- Cantidad de Generaciones



Otras Variantes:

- **Mutación Probabilística/Determinística**: se aplica el operador de Mutación a TODA la población, pero se realiza la mutación con una baja probabilidad (usualmente 1 o 2 %), o se realiza la mutación sobre un 2 % de la población elegido al azar (pero porcentaje fijo).
  - En el 1º caso, la variación de los generadores aleatorios hará que algunas veces no se mute ninguna solución, y otras hasta el 4 % (similar al caso de la ruleta). En cambio la aplicación determinística asegura que se realizará el número prefijado de mutaciones, todas las veces.

  - Esto mejora la REPETIBILIDAD del algoritmo (estabilidad por repeticiones, qué ocurre cuando se larga el AG nuevamente)
- Problema del Crossover repetido: Muchas veces el operador de Crossover se aplica sobre 2 padres idénticos (lo cual no genera nada nuevo y gasta recursos), especialmente sobre el final de la corrida donde la mayoría de la población son muy similares entre sí.
- Incluso en el caso de 2 padres no exactamente iguales, la elección desafortunada de uno o mas puntos de corte puede hacer que se generen hijos iguales a los padres. Se puede modificar los operadores de Crossover de modo que generen hijos distintos a sus padres, siempre que sea posible.
- En general esto se soluciona introduciendo una permutación al azar (shuffle) luego de la selección y antes de dividir en parejas (pairing). En algunos casos esto no resulta suficiente, y existen estrategias que intentan asegurarse que los padres sean distintos entre sí (Sin embargo esto No Es siempre posible).

Codificación del Problema:

- Basadas en el Orden: el orden en que aparecen las variables es importante.
  - Permutaciones (TSP)
  - Otras (Sudoku)
- No Basadas en el Orden: el orden no es importante, sino el valor de cada variable.
  - Codificación Binaria (problema de los subconjuntos)
    - Común
    - Códigos de Gray
  - Codificación a Valores Finitos: las variables sólo pueden tomar una cantidad finita y prefijada de valores, normalmente clases. Por ej. Colores disponibles, coeficientes de un polinomio de grado dado, etc.
  - Entera
  - Real
- Codificación Directa: el vector solución representa una solución en la realidad, sin necesidad de procesos de Traducción, o si existe, es mínimo. O sea, contiene toda la información necesaria para implementar la solución en la realidad. Ej: Regresión Polinomial.
- Codificación Indirecta: el vector solución no representa directamente una solución en la realidad. Requiere el acceso a datos externos y un proceso de traducción no trivial para producir una solución real. Ej: TSP en grafos (acceso a la matriz de adyacencia del grafo, información sobre las ciudades representadas, traducción de la forma vectorial a las aristas del grafo)

Usualmente la codificación Indirecta es computacionalmente más costosa (en memoria y tiempo de procesamiento), pero es mucho más efectiva, ya que contiene más conocimiento sobre el problema

Elitismo en los Algoritmos Genéticos:

Dentro del cómputo evolutivo existen diferentes paradigmas. Aunque todos los paradigmas se basan en la misma idea del neo-darwinismo y en el uso de una población de soluciones, difieren entre ellos por la forma de implementar los mecanismos de selección, apareamiento, mutación y elitismo.

Los principales paradigmas son: los algoritmos genéticos, las estrategias evolutivas y la programación evolutiva.

También existen otras técnicas como la programación genética, la evolución diferencial, la optimización mediante cúmulos de partículas, la optimización por colonia de hormigas, los algoritmos culturales, los sistemas inmunes artificiales y la búsqueda dispersa.

En la actualidad, el paradigma de algoritmos genéticos se presume como el más popular.
En cuanto a la representación, tradicionalmente se hace uso de una cadena binaria, es decir, trabajan a nivel de genotipo, lo que equivale a transformar el problema de un espacio cualquiera al binario. Por ello, en el algoritmo genético una representación adecuada es un factor importante para obtener buenos resultados.

Tradicionalmente, los algoritmos genéticos hacen uso de la selección proporcional con base en la adaptabilidad. Le da mayor importancia al operador de apareamiento que al de mutación, y no cuentan con mecanismos de auto-adaptación. En cuanto a la probabilidad de apareamiento se utilizan valores altos, al contrario de la probabilidad de mutación donde los valores usualmente son bajos.
Como ya observamos los algoritmos genéticos son técnicas de optimización que se basan en la evolución de las especies, son utilizados especialmente en problemas bastante complicados donde las técnicas tradicionales de optimización no obtienen buenos resultados.

El pensamiento evolutivo actual gira en torno al Neo-Darwinismo, que está basado principalmente en las ideas de Darwin, Weismann y Mendel, el cual establece que toda la vida en el planeta puede ser explicada a través de: selección, apareamiento, mutación y competencia.

En sí, en los algoritmos evolutivos se requiere codificar las estructuras del problema de optimización, definir las operaciones entre los individuos, una función de adaptabilidad y los mecanismos de selección. Los algoritmos genéticos inicialmente se conocieron como planes reproductivos, y fueron introducidos por John H. Holland a principios de los años sesenta y fueron utilizados en el aprendizaje automático. El algoritmo genético es un algoritmo evolutivo en el cual el apareamiento es el operador principal y la mutación es un operador secundario.

Coloquialmente, por élite se entiende un grupo pequeño que por algún motivo, característica, facultad o privilegio es superior o mejor en comparación al grueso de una población determinada; con cualidades o prerrogativas de las que la gran mayoría no disfrutan.

En general, se habla de élite como sinónimo de elegido, escogido, eminente o distinguido. Esta concepción tiene más o menos el mismo significado con que éste término es manejado en las ciencias sociales. Es común también que se le llame elitista a quienes son selectivos, a quienes discriminan a otros, a los que manifiestan repulsión por lo común o popular, que le dan una valoración negativa o califican peyorativamente a los conceptos de masa y mayoría.

Un algoritmo genético, desde el punto de vista de la optimización, es un método poblacional de búsqueda dirigida basada en probabilidad. *Bajo una condición bastante 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 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.
El método más utilizado para mejorar la convergencia de los algoritmos genéticos es el elitismo. Este método consiste básicamente en realizar la etapa de selección en dos partes:

(1) Se realiza un muestreo en una élite de "ere" miembros de entre los mejores de la población inicial y se incorporan directamente a la población final, sin pasar por la población intermedia.
(2) (2) La población auxiliar de criadores se muestrea de entre los "total menos ere" restantes miembros de la población inicial. Normalmente, el tamaño de la élite "ere" es bastante pequeño y el tipo de muestreo es bien directo o bien por sorteo, ambos en la variedad diversa.

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) La definición de algunos operadores genéticos que, al actuar sobre una o varias soluciones, suministren una o más soluciones al alterar genéticamente los cromosomas. Juegan el papel del apareamiento y la mutación en el proceso evolutivo natural.

En principio aparenta una cierta conveniencia contar con una estrategia de selección estricta para que mejore rápidamente la población y converja el algoritmo, es decir, que la población se acumule alrededor de un genotipo óptimo. Esto no es cierto. Lo que ocurre es que la población se acumula rápidamente alrededor de algún individuo que es bueno, comparativamente con el resto de los individuos considerados a lo largo de la ejecución del algoritmo, pero este individuo puede no ser el mejor posible. A esto se le suele llamar convergencia prematura. No se puede asegurar pero sí procurar que lo anterior no ocurra.

Además de la explotación es necesario que exista exploración. El algoritmo genético debe, no sólo seleccionar de entre lo mejor que ha encontrado, sino procurar encontrar mejores individuos. En la estrategia de selección normalmente se incluye un elemento extra que sirve de "ancla". Si sólo se hace selección forzando que sea más probable elegir al mejor individuo de la población pero sin asegurarlo, es posible que este individuo se pierda y no forme parte de la siguiente generación. Para evitar lo anterior *se fuerza la selección de los mejores individuos de la generación para pasar intactos a la siguiente*. A esta estrategia se le denomina elitismo y puede ser generalizada especificando que permanezcan en la población los mejores individuos de las pasadas generaciones.

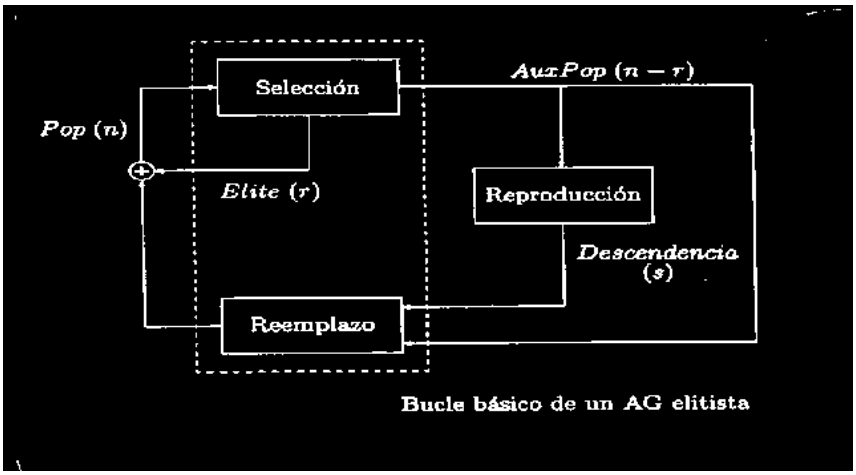
El mecanismo elitista pretende asegurar que aquel o aquellos individuos que son los más aptos de la población actual sobrevivan y continúen participando en el proceso evolutivo, pasando a la siguiente generación de manera intacta, sin recombinarse ni mutarse.
Implementar este mecanismo asegura que la mejor aptitud encontrada hasta el momento, el mejor individuo hasta el momento, no se perderá en la siguiente generación.

El elitismo es importante, pues garantiza, a través de pruebas matemáticas, la convergencia global del algoritmo genético. Se acepta que con el elitismo, cuando se mantiene una copia del mejor individuo es posible garantizar convergencia global para un problema de optimización. En este dominio existen variantes de modelos elitistas a las que se denominan elitismo parcial y elitismo total. Por elitismo parcial se quiere decir que en una población de tamaño "ene" se mantiene una copia de los mejores "eme" individuos hasta el total de generaciones definidas. Por elitismo total se quiere decir que se mantiene una copia de los mejores "ene" individuos, el total de individuos de la población, hasta el total de generaciones definidas.

Bajo ciertas condiciones muy generales, la introducción del elitismo garantiza la convergencia teórica al óptimo global; en la práctica, mejora la velocidad de convergencia de los algoritmos genéticos cuando la función de evaluación es unidmodal, es decir, no hay subóptimos, sin

embargo la velocidad de convergencia empeora con funciones fuertemente multimodales. Con tamaños de población pequeños se consiguen efectos similares a los del elitismo introduciendo un proceso de reinicialización periódica en los algoritmos genéticos: cada vez que el algoritmo genético converge se salvan los mejores individuos, se reinicializan los demás y se vuelve a comenzar. La reinicialización tiene efectos beneficiosos sobre las prestaciones del método debido a que introduce diversidad, requisito especialmente crítico en los algoritmos genéticos con poblaciones pequeñas.

Al considerar la Selección Elitista debemos tener en cuenta que en ciertas ocasiones puede suceder que tras el cruce y la mutación, perdamos el cromosoma con mejor adaptación. Para evitar lo anterior, el método de selección elitista, copia el mejor cromosoma o alguno de los mejores en la nueva población. El resto se realiza de la misma forma que se trabajaba.
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
El esquema básico de un algoritmo genérico elitista puede ser como sigue:



Índice de Contenidos**:**

Población 1
Tamaño de la población 1
Población inicial 1
Función objetivo 1
Selección 2
Cruce 4
Mutación 8
Operadores de Mutación 9
Arquitectura de un Algoritmo Genético 16
Elitismo en los Algoritmos Genéticos: 18
Bibliografía 22

Bibliografía

Artículo: “Algoritmos Genéticos y Optimización Heurística” por el Dr. Adrian Hill. Grupo de Aplicaciones de Inteligencia Artificial. Universidad Nacional de Tucumán
awill@herrera.unt.edu.ar

Artículo desarrollado por Guillermo Choque Aspiazu. Marzo 15 de 2010
http://www.eldiario.net/

Material extraído de : http://geneura.ugr.es/~jmerelo/DegaX/GenAlg.html