LOS ALGORITMOS GENETICOS EN LA OPTIMIZACIÓN DE FUNCIONES

Ing. Alejandro Hernández Hernández

Universidad de Matanzas – Sede "Camilo Cienfuegos",Vía Blanca Km.3, Matanzas, Cuba. alejandro.hdez@umcc.cu

CD de Monografías 2017

ISBN: XXX-XXX-XX-XXXX-X

Resumen

Actualmente los Algoritmos Genéticos son una opción muy atractiva como herramienta para la solución de problemas complejos de búsqueda y optimización, debido a su probada capacidad de encontrar soluciones óptimas en un tiempo computacionalmente aceptable y además resultan ser de las pocas técnicas efectivas en problemas que antes se resolvían gracias a la intuición y experiencia de los investigadores. Esta optimización abarca parámetros tales como son recursos, materiales, tiempo de ejecución, factibilidad, costos entre otros lo que facilita mejoras no solo en la calidad sino también en la economía. En la presente investigación se analizó las principales características y funcionamiento de los algoritmos genéticos, y se obtuvo un documento guía para la posterior profundización en la aplicación de Algoritmos Genéticos en la optimización de problemas, el mismo puede ser utilizado como una base para el estudio y aplicación en múltiples ramas de la ingeniería.

El resumen tendrá entre 100 y 150 palabras, en un solo párrafo. Explicará brevemente el objetivo del trabajo, los principales resultados y conclusiones. Se evitará el uso de símbolos y abreviaturas.

A continuación, se indicarán no más de seis palabras claves que identifiquen la temática tratada.

***Palabras claves**: Algoritmos Genéticos, Problemas Multiobjetivos, Soluciones Óptimas*.

Introducción

Los Algoritmos Genéticos surgen como herramientas para la solución de problemas complejos de búsqueda y optimización, producto del análisis de los sistemas adaptativos en la naturaleza, y como resultado de abstraer la esencia de su funcionamiento. El término Algoritmo Genético se usa por el hecho de que estos simulan los procesos de la evolución darwiniana a través del uso de operadores genéticos que operan sobre una población, de individuos o cromosomas, que "evoluciona" de una generación a otra. Los algoritmos genéticos han ganado popularidad en los últimos años debido a la posibilidad de aplicarlos en una gran gama de campos y a las pocas exigencias que imponen al problema a resolver. Han mostrado además bondades en la resolución de problemas multiobjetivos. Este nombre se le atribuye a Holland, sin embargo, otros científicos trabajaron en ideas parecidas durante la década del sesenta del pasado siglo En particular, Ingo Rechenberg y Hans-Paul Schwefel, en Alemania, desarrollaron las ideas de las estrategias evolutivas, mientras que Bremermann y Fogel desarrollaron las ideas de programación evolutiva. La parte común en estas estrategias es la mutación y la selección de los operadores genéticos que se estudiaran posteriormente.

A diferencia de las otras estrategias evolutivas, los algoritmos genéticos se concentran más en la combinación de soluciones y permite tener una representación codificada de las variables del problema, en lugar de las variables mismas.

ISBN: XXX-XXX-XX-XXXX-X

Los principios básicos para la implementación de un algoritmos genéticos son: la función de adaptación (Fitness), la representación de las soluciones (codificación de los individuos de la población), el esquema de reemplazo (determina en cierto modo el tipo de algoritmo genético), los operadores (selección, cruce y recombinación, mutación). Los Algoritmos Genéticos tienen como campo de aplicación la resolución de problemas diversos, pero estos se especializan en problemas de alta complejidad, aplicándose en disímiles áreas del conocimiento.

La principal diferencia de los algoritmos genéticos con otras metaheurísticas de optimización es la utilización de conjuntos de soluciones de forma simultánea durante la búsqueda.

Por otra parte numerosos problemas a los que se enfrentan los ingenieros civiles están descritos por la optimización de estructuras. En este trabajo se propone el uso de los algoritmos genéticos para evaluación de estructuras diversas.

Desarrollo

Bases de la computación evolutiva. La computación evolutiva nace como consecuencia de los inconvenientes y limitaciones de los métodos más formales como la Programación Matemática al ser aplicados a problemas complejos. Estas técnicas tradicionales requieren de funciones continuas y derivables. Tal situación no siempre es posible cuando se abordan problemas de optimización. Fue en las décadas de 1950 y 1960 cuando varios científicos, de modo independiente, la idea era "evolucionar" una población de candidatos a ser solución de un problema conocido, utilizando operadores inspirados en la selección natural y la variación genética natural. Entre los años 1960 y 1970, un conjunto de investigadores de entre los que destacan Lawrence J. Fogel, Ingo Rechenberg, y John Henry Holland, empezaron de forma teórica a plantear la posibilidad de desarrollar algoritmos de optimización inspirados en la naturaleza. Fue el último de estos el que acuñó por primera vez el término Algoritmo Genético.

En términos muy generales se podría definir la computación evolutiva como una familia de modelos computacionales inspirados en la evolución. Más formalmente, el término de computación evolutiva se refiere al estudio de los principios de la evolución natural (Alba and Tomassini, 2002). Estas técnicas heurísticas podrían clasificarse en 3 grandes categorías o grupos, los algoritmos genéticos, las estrategias de evolución y la programación evolutiva.

Los Algoritmos Genéticos (AG): Fueron desarrollados por John H. Holland y sus colaboradores. Utilizaron inicialmente la codificación binaria, aunque en la actualidad también se ha extendido a la codificación con números reales. Emplea operadores genéticos de selección, recombinación y mutación teniendo mayor relevancia los dos primeros operadores. Para la formación de los descendientes la reproducción es sexual, es decir, hay intercambio del material genético de los padres mediante los operadores genéticos, cruce y mutación. En este caso, el operador mutación se considera un operador de segundo orden frente al operador cruce, como principal diferencia con las técnicas evolutivas anteriores(Holland, 1962).

ISBN: XXX-XXX-XX-XXXX-X

La Programación Evolutiva (EP): Fue desarrollada por Lawrence J. Fogel con la idea de simular la evolución como medio de aprendizaje con el fin de obtener una inteligencia artificial. Esta trabaja directamente con las variables de diseño y los nuevos individuos son generados a partir de un solo padre mediante el uso exclusivo de la mutación. En cierto modo es una variación de los Algoritmos Genéticos con reproducción asexual. Emplean solamente los operadores genéticos de mutación y selección(Fogel, 1967).

Las Estrategias Evolutivas (ES): Fueron desarrolladas por Ingo Rechenberg, Hans-Paul Schwefel y sus colaboradores. Utilizan generalmente la codificación con números reales. La función de codificación transcribe dos tipos de variables: las variables objeto y las estratégicas. Las variables objeto se corresponden con las del problema que se desea resolver mientras que las estratégicas son los parámetros mediante los cuales se gobierna el proceso evolutivo. Emplea operadores genéticos de selección y mutación (Rechenberg, 1973, Schwefel, 1977).

Estas tres áreas, estrategias evolutivas, algoritmos genéticos, y programación evolutiva forman la columna vertebral de la Computación Evolutiva, y de ellas parten los caminos hacia todos los campos de investigación inspirados en los conocimientos sobre evolución (Ai and Wang, 2011, Wang and Arora, 2004, Wang, 1991).

Desde un punto de vista general las técnicas de Computación Evolutiva, como los Algoritmos Genéticos, pueden considerarse como un conjunto de técnicas computacionales más ligadas en sus conceptos a los procesos biológicos que a las técnicas computacionales tradicionales. Es por ello que este tipo de técnicas en ocasiones puede convertirse en un camino largo plagado de inconvenientes en el que se debe lograr una correcta interrelación entre los conceptos básicos del mundo biológico y el mundo computacional para que se logre demostrar el gran potencial que ofrecen estas técnicas.

En estos últimos años se ha generado una amplia interacción entre los investigadores de varios métodos de computación evolutiva, rompiéndose las fronteras entre algoritmos genéticos, estrategias evolutivas y programación evolutiva. Como consecuencia, en la actualidad, el término “algoritmo genético” se utiliza para designar un concepto mucho más amplio del que concibió Holland con anterioridad.

Características de la computación evolutiva. Todas las técnicas de la computación evolutiva se basan en el modelo de la evolución natural. Se utiliza una estrategia de aprendizaje colaborativo a partir de un conjunto de individuos. Normalmente, cada individuo representa o codifica un punto del espacio de búsqueda de soluciones en un problema dado. Cada individuo incorpora información adicional que permite llegar a la solución final del problema a partir de descendientes. La descendencia, se genera de forma pseudo-aleatoria mediante procesos de cruce y recombinación donde se intercambia información entre dos o más individuos actualmente existentes. La mutación es un proceso de auto-replicación errónea de los individuos que produce pequeños cambios en los mismos. Mediante la evaluación de los individuos, se consigue una medida de la adaptación de cada uno de ellos a su entorno. De acuerdo con esta medida de adaptación, el proceso de selección favorece más a los individuos mejor adaptados, que serán por tanto los que se reproduzcan más frecuentemente(Goldberg, 1989).

ISBN: XXX-XXX-XX-XXXX-X

Estas son las características generales que comparten todas sus ramas. Sin embargo, existen diferencias de implementación de estos principios que caracterizan y diferencian a los algoritmos genéticos, las estrategias evolutivas y la programación genética (Back et al., 1991, Back and Schwefel, 1993).

Las estrategias evolutivas normalmente usan la mutación para modificar vectores de números reales, empleando la recombinación (cruce) como operador principal de búsqueda. El operador de selección es determinístico y, normalmente, el tamaño de la población de los padres y el de la descendencia es distinto.

Los algoritmos genéticos se basan en el cruce como el operador de búsqueda más importante y el operador de mutación se aplican con una probabilidad muy pequeña, y no es considerado un operador fundamental. Se usa la selección probabilística y, generalmente, se emplea la codificación de los individuos mediante cadenas binarias, aunque también se pueden emplear otras codificaciones tales como las basadas en números reales.

La programación genética es una extensión de los algoritmos genéticos. Busca construir programas de computador sin que ellos sean diseñados y programados expresamente por un humano. Puede decirse que es una técnica de optimización cuyo espacio de búsqueda son los posibles programas de computador que solucionan un problema determinado.

Historia de los algoritmos evolutivos en particular de los algoritmos evolutivos genéticos.

Los Algoritmos Evolutivos incluyen a todas las técnicas relacionadas con la optimización metaheurística, estas surgieron por el año 1960. Con el auge que tomaron estas teorías, y por medio de los descubrimientos de nuevas tecnologías que permitían un desarrollo computacional y matemático mucho más elevado, empezaron a investigarse teorías y algoritmos artificiales sentando sus bases en los mecanismos naturales siendo una fecha importante 1975 con los trabajos de Holland (Fraser, 1958, Fraser, 1957, Holland, 1975).

A partir de esta fecha, los estudios sobre algoritmos evolutivos, en cualquiera de sus facetas, se han incrementado de tal forma que ya es difícil encontrar un área de investigación en la que no hayan irrumpido con gran fuerza. Se han encontrado implementados en ámbitos tan diversos como finanzas, inversión, educación, transporte, redes de comunicación, planificación, logística, producción, optimización, etc.(Biethahn and Nissen, 1995).

Las primeras ideas en las que se basan los algoritmos genéticos se pueden encontrar en los artículos de Holland de principios de los años 1960. A mediados de los años 1960, las ideas de Holland empezaron a plasmarse en modelos implementados en ordenador. En cada uno de estos sistemas se representaban elementos por medio del uso de genomas en los que los mecanismos de evolución y herencia eran abstracciones de operadores genéticos como la mutación, el cruce o la inversión. Estos estudios experimentales desembocaron en el tratamiento de problemas de búsqueda más complejos, como el reconocimiento de patrones. Asimismo, se realizaron los primeros estudios con operadores de selección de tipo elitista y las ideas básicas para el empleo de probabilidades adaptativas de mutación y cruce. También en esta época se estudió detalladamente, por primera vez, las diferentes posibilidades de reproducción y cruce. Usando dos espacios de soluciones diferentes, se experimentó con una amplia variedad de operadores de cruce.

ISBN: XXX-XXX-XX-XXXX-X

En paralelo con estos estudios experimentales, Holland continuó trabajando en una teoría general de los sistemas adaptativos. Muchas de las propiedades de los algoritmos genéticos identificadas por Holland de forma teórica no pudieron ser observadas experimentalmente debido a la falta de recursos computacionales. Esto obligaba a trabajar con un número limitado de generaciones y con un número muy pequeño de individuos, habitualmente menos de 20(Holland, 1975).

Estos estudios fueron ampliados posteriormente de forma teórico-práctica con el análisis de los efectos cruzados de la modificación del tamaño de la población, cruce y mutación en el comportamiento de una familia de algoritmos genéticos utilizados para optimizar un conjunto fijo de funciones de prueba. A partir de estos estudios se empezó a manifestar el enorme potencial de los algoritmos genéticos para resolver problemas de optimización (Jong, 1975, Coello, 1999).

Dado el creciente interés en los algoritmos genéticos, se organizó en 1985 el primer "International Congress on Genetic Algorithms" donde se presentaron los últimos avances tanto teóricos como prácticos en el empleo de estas técnicas. El éxito de esta edición del congreso consiguió que se repitiera su celebración cada dos años. A partir de 1989, las actividades en el campo de los algoritmos genéticos habían crecido mucho, por lo que se fundó la "International Society for Genetic Algorithms (ISGA)"(Greffenstette, 1986, Greffenstette and Baker, 1989).

Fue precisamente David Goldberg, un discípulo de Holland quien alcanzo mayor éxito en la temática de optimización. Según parece tras asistir a uno de los seminarios de Holland, Goldberg que estaba interesado en encontrar el diseño óptimo de líneas para el transporte de gas, le planteo la posibilidad de emplear los Algoritmos Genéticos con este fin. Poco tiempo después logro llevar a cabo dicha tarea en su tesis doctoral. Posteriormente publicaría un libro que se convertiría un hito en el campo de la optimización y que recogía numerosas aplicaciones, muchas de ellas relacionadas con la optimización de estructuras, como el clásico problema de optimización de diez barras y seis nudos, aunque sin la restricción de desplazamientos del problema original(Goldberg, 1989).

Posteriormente en 1992 se empleó un Algoritmo Genético binario para optimizar una estructura de tres barras y la clásica estructura de diez barras pero añadiendo las restricciones de desplazamiento del problema original. También aplicaron el algoritmo para optimizar una torre de transmisión de 160 barras. Ese mismo año se utilizó un Algoritmo Genético binario que empleaba variables discretas para minimizar una estructura de diez barras sujeta a restricciones de desplazamiento. Al año siguiente utilizaron un Algoritmo Genético con dos estrategias de cruce diferentes para optimizar estructuras a gran escala, aplicando el algoritmo a estructuras de 25 y 72 barras. En este mismo se implementó un Algoritmo Genético binario para optimizar el tamaño y la topología de diversos pórticos. Este es uno de los primeros trabajos en analizar estructuras con elementos de viga. En 1994 el aporte se manifestó en la aplicación de un algoritmo a estructuras tridimensionales(Jenkins, 1991, Jenkins, 1997, Jenkins, 2002, Jenkins, 1992, RAJEEV and KRISHNAMOORTHY, 1992, Rajeev and Krishnamoorthy, 1997, Hajela and Lin, 1992, Grierson and Pak, 1993, Ai and Wang, 2011, Hajela and Ashley, 1981).

ISBN: XXX-XXX-XX-XXXX-X

La tendencia ha sido la de un tremendo crecimiento y diversificación del área de investigación de los algoritmos genéticos, como se refleja en el éxito de las distintas conferencias y la proliferación de libros y publicaciones periódicas. La interacción de los algoritmos genéticos con otras técnicas de la Computación Evolutiva se ha traducido en un muy productivo intercambio de ideas y la aparición de nuevas aplicaciones híbridas.

Algoritmos genéticos.

La inteligencia artificial como rama de la Ciencia de la Computación se encarga de simular comportamientos inteligentes en los ordenadores. Los algoritmos genéticos son, a su vez, una rama de la inteligencia artificial basada en la teoría de la evolución de Darwin por lo que se pueden definir como métodos adaptativos que pueden utilizarse para resolver problemas reales de búsqueda y de optimización. Este tipo de algoritmos, se utiliza para abordar una amplia variedad de problemas en un conjunto de campos sumamente diverso, demostrando claramente su capacidad y su potencial(Baker, 1987, Syswerda, 1991).

Un algoritmo es un conjunto ordenado y finito de operaciones que permite ejecutar un procedimiento o resolver un problema, un algoritmo es genético cuando utiliza principios de la genética y la selección natural (Ai and Wang, 2011).Estos trabajan sobre estructuras de cadena que son una representación de una posible solución al problema denominado cromosomas para comenzar, se inicializa la población completamente al azar. En la inicialización hay que tener en cuenta que la distribución de valores debe ser uniforme para cada rango representado por los cromosomas, estos evolucionan a través del tiempo de acuerdo a reglas de supervivencia que privilegian a los que presenten mejor desempeño dándoles una mayor probabilidad de que a partir de ellos se cree la siguiente generación de acuerdo con la función de adaptación; del mismo modo que en la naturaleza los individuos mejor dotados (es decir, los que ofrezcan las mejores soluciones) serán los que más probabilidades tendrán de ser seleccionados para reproducirse. Mientras que los individuos peor adaptados (los que ofrezcan peores soluciones) tendrán más difícil el propagar su material genético a las nuevas generaciones. Sucesivamente cada nueva generación contará con una mayor proporción de buenas características de forma que, si el algoritmo genético ha sido diseñado correctamente, la población convergerá hacia una solución óptima del problema (Fraser, 1958, Bremermann, 1962).

En algunos casos para que un elemento de la población pueda pasar a través de la función de aptitud debe ser codificado, este proceso de codificación es clave para permitir que el algoritmo converja de manera más eficiente hacia la solución del problema planteado, en principio la codificación puede realizarse con cualquier alfabeto finito, sin embargo, no todas las codificaciones resultan ser computacionalmente eficientes, es deber del programador dependiendo de la plataforma de software o hardware que elija escoger que codificación resulte más conveniente (Coello et al., 2007, Goldberg, 1989, Jong, 1975).

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

Los Algoritmos genéticos tienen capacidades importantes en la rama de la optimización, entre ellas: Son métodos iterativos que convergen a partir de un conjunto inicial y arbitrario

ISBN: XXX-XXX-XX-XXXX-X

de puntos del espacio de búsqueda. La convergencia no está condicionada por requisitos de continuidad o diferenciabilidad del funcional a maximizar o minimizar. En su convergencia al óptimo global escapan de la atracción de múltiples óptimos locales. Son aplicables y eficientes en la optimización multicriterio.

Los algoritmos genéticos suelen ser eficaces con funciones no derivables, y hay que tener en cuenta que si existen muchos máximos/mínimos locales se necesita de más iteraciones para asegurar el local, y que si tiene muchos puntos cercanos al óptimo, no se puede asegurar que se encontrará dicho óptimo, pero sí al menos uno de esos puntos. Esta es una de sus mayores desventajas, aunque se ha demostrado que sí que encuentran soluciones muy buenas en un tiempo realmente breve en referencia a otros sistemas de búsqueda de soluciones.

Una definición bastante completa de un algoritmo genético es la propuesta por 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 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 " (Koza, 1992).

Óptimo local, global y de Pareto.

Óptimo local: Se define el óptimo local $x^* \in X$ de una función $f: X \rightarrow R$ como aquel elemento que es máximo o mínimo local de la función.

Óptimo global: Se define el óptimo global $x^* \in X$ de una función $f: X \rightarrow R$ como aquel elemento que es máximo o mínimo global de la función(López, 2010).

Óptimo de Pareto: Este término aparece cuando se afronta problemas de optimización de más de una variable y con restricciones. En estos casos no todas las soluciones son viables. Y más aún, los óptimos de Pareto hacen referencia a soluciones en las que no se puede mejorar más los valores de una variable, sin empeorar alguna de las otras condiciones. Es decir que más que alcanzar una solución única, lo que se consigue es definir una frontera de soluciones equilibradas (frente de Pareto) donde no es posible encontrar una solución que mejore en ningún sentido sin empeorar en otro(Coello et al., 2007).

Optimización.

Se trata de un campo especialmente abonado para el uso de los Algoritmos Genéticos, se han utilizado en numerosas tareas de optimización, incluyendo la optimización numérica, y los problemas de optimización combinatoria (Goldberg, 1989, Coello, 1999, Baker, 1987, Syswerda, 1989).De forma genérica, puede definirse la optimización como aquella ciencia encargada de determinar las mejores soluciones a problemas matemáticos que a menudo modelan una realidad física(López, 2010, Coello, 1999).La optimización de varias funciones objetivos involucrada donde la suerte favorece al decidir qué solución es la correcta(Coello et al., 2007).

Optimización con restricciones.

ISBN: XXX-XXX-XX-XXXX-X

En el caso de que se esté trabajando en un problema de optimización con restricciones se puede ver que no todas las soluciones posibles son aceptables. Esto implica que previamente se ha modelado dicho problema de manera que se pueda trabajar como una función matemática, cosa que resulta muy complicada en algunos casos. En muchas ocasiones el problema a optimizar no tiene una representación matemática directa, sino que se trabaja con algún tipo de función objetivo o incluso con simulaciones de los resultados(López, 2010).

Espacio de búsqueda.

El espacio de búsqueda se define como el conjunto de todos los elementos que pueden ser operados por los métodos de optimización. Esta diferenciación cobra especial sentido en los algoritmos genéticos y en la mayoría de los algoritmos heurísticos ya que no trabajan con soluciones directamente, sino con individuos codificados que representan las posibles soluciones. La búsqueda generalmente se realizará con los mejores individuos de distintas áreas del espacio del problema (López, 2010, Coello, 2014)

Para resolver computacionalmente el problema de minimizar funciones utilizando algoritmos genéticos se utilizan intervalos pequeños con el objetivo de no hacer demasiado grande el espacio de búsqueda del problema. A mayor tamaño del espacio de búsqueda del problema se necesita una mayor cantidad de iteraciones del algoritmo para obtener soluciones satisfactorias (Thomas, 2009).

Es evidente que el tamaño del espacio de búsqueda depende de la precisión con que se desee obtener las soluciones, esta podrá ser seleccionada por el usuario de la aplicación.

La vecindad se define como un entorno acotado del espacio de búsqueda alrededor del punto que se está evaluando donde todos los puntos son adyacentes(López, 2010, Thomas, 2009).

Tamaño de población.

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 se debe seguir una serie de pasos. 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 se debe tener en cuenta que se garantice una cierta diversidad en la población generada.

Una cuestión que debe plantearse es la relacionada con el tamaño idóneo de la población para garantizar la diversidad de soluciones al aplicar algoritmo genético en el campo de la resolución de problemas. Las poblaciones pequeñas corren el riesgo de no cubrir adecuadamente el espacio de búsqueda, mientras que al trabajar con poblaciones de gran tamaño puede acarrear problemas relacionados con el excesivo costo computacional (Goldberg, 1989, Jong, 1975, Alander, 1992).

Aunque los algoritmos genéticos son eficientes, sin embargo no garantizan la obtención de una solución óptima. Su efectividad viene claramente determinada por el tamaño de la población. Por eso se debe buscar un compromiso entre el número de individuos utilizados y la calidad que se desea alcanzar. De ahí se pudiera destacar la importancia de utilizar poblaciones múltiples para aumentar la diversidad de las soluciones.

Las poblaciones múltiples pueden hacer búsquedas en regiones que el algoritmo estima provechoso, eliminando parte del espacio de búsqueda, lo que dificultaría que se pudiera

ISBN: XXX-XXX-XX-XXXX-X

encontrar la solución óptima en los problemas engañosos, debido a que la presión de selección es demasiada alta(Coello et al., 2007, Baker, 1987, Syswerda, 1989).
La población se encuentra particionada en algunas subpoblaciones de menor tamaño que evolucionan independientemente unas de otras, y que intercambian información con una frecuencia determinada de forma que los individuos pueden interactuar con sus vecinos más próximos en la población(Alba and Tomassini, 2002)(Díaz, 2006, Wang and Arora, 2004, Coello et al., 2007, Jong, 1975, Smith, 1980).
En lugar de ejecutar algoritmos genéticos con muchas generaciones, es mejor compilar los programas muchas veces con una cantidad menor de generaciones y creando poblaciones iniciales por medio de resultados históricos, de esta manera se garantiza que cada vez que se ejecute el programa se encontrarán soluciones finales mejores o por lo menos iguales a la compilación anterior (Calderon, 2005).

Función objetivo o función evaluación.

Los algoritmos genéticos requieren funciones para evaluar el estado o valor al decodificar sus genotipos, existen dos funciones usadas con este fin y son definidas por Coello como: Función objetivo: Define la optimización del algoritmo genético (tiene que ver con el problema matemático en sí). Función de adaptación: Mide qué tan bien una solución particular satisface una condición y asigna un valor real a su solución (relacionado con el algoritmo).Darwin definió la función de adaptación para miembros de una población como "su habilidad para sobrevivir a los depredadores, la pestilencia y otros obstáculos que interfieran en su crecimiento y reproducción" (Coello, 2014).
La función evaluación juega un papel importante en la clasificación potencial de las soluciones en términos de sus características; es el criterio de optimización y evaluación de la calidad de los individuos. Se desarrolla a partir de los valores del fenotipo (Thomas, 2009, Davis, 1991).
En el caso en que la computación de la función objetivo sea muy compleja se emplea 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 (Chau and Albermani, 2003, Thierens and Goldberg, 1994).

Función de adaptación o Fitness.

En la naturaleza, los individuos más adaptados son los que tienen más oportunidades de reproducirse y de que su material genético se propague de generación en generación. Los algoritmos genéticos intentan imitar este proceso y es la función de adaptación o Fitness la encargada de evaluar cómo de "buenos" son los individuos (Bremermann, 1962).
Existen multitud de formas de evaluar a los individuos de una población: desde las más simples como la de asignar un valor real que defina cómo de bueno es el individuo comparado con el mejor encontrado hasta la fecha (Fitness Puro), hasta normalizar el valor entre 0 y 1 en función de los valores obtenidos por todos y cada uno de los individuos (Fitness Normalizado). Además de tener en cuenta posibles penalizaciones puesto que no todos los individuos pueden ser soluciones viables. Pueden incumplir alguna de las restricciones propias del problema y no por ello han de ser descartados sin más aunque sería una opción posible.

ISBN: XXX-XXX-XX-XXXX-X

La función de adaptación considera las medidas clásicas de precisión y exhaustividad ya que es la encargada de modelar el entorno, definiendo la calidad de las soluciones según el problema (Koza, 1992, Eiben and J.E.Smith, 2007, Rajan, 1995).

Función de penalización.

Las funciones de penalización constituyen uno de los métodos de manejo de las restricciones más empleados. La idea fundamental del método consiste en transformar un problema restringido en uno sin restricciones añadiéndole (o restándole) un determinado valor a la función objetivo dependiendo del grado de violación de las restricciones presentes en una determinada solución.

En los métodos de optimización existen dos tipos diferentes de funciones de penalización: las exteriores y las interiores. En las primeras se parte soluciones localizadas en el espacio de soluciones no factibles y desde aquí el algoritmo va desplazando las soluciones hacia el espacio de soluciones factibles. Por el contrario en las segundas se parte de soluciones factibles eligiendo un factor de penalización que es muy pequeño y que crece hasta el infinito en la frontera entre el espacio de soluciones factibles y no factibles. De este modo las restricciones actúan como barreras del proceso de optimización.

El método más empleado en los algoritmos genéticos, y especialmente en la optimización de estructuras, es la penalización exterior ya que en muchos casos es muy difícil partir de una solución factible, así como generar nuevos individuos factibles a partir de las poblaciones iniciales (Caballero, 2012, Ai and Wang, 2011, Back et al., 1991, Adeli and Cheng, 1993, Hajela and Lee, 1993).

Convergencia.

Para criterios prácticos, es muy útil la definición de convergencia introducida en este campo por De Jong. Si el Algoritmo Genético ha sido correctamente implementado, la población evolucionará a lo largo de las generaciones sucesivas de tal manera que la adaptación media extendida a todos los individuos de la población, así como la adaptación del mejor individuo se irán incrementando hacia el óptimo global. 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(Jong, 1975, Thomas, 2009, Koza, 1992).

Se dice que un gen converge cuando el 95% de la población comparte el mismo valor para ese determinado gen, o bien que todos los valores que toma ese gen en los individuos de la población sufren variaciones de no más del 5% para el caso de alfabetos infinitos. Asimismo, se dice que la población converge cuando todos los genes han convergido(Coello et al., 2007).

Hay que tener en cuenta que tanto la convergencia de los algoritmos genéticos, como su velocidad de convergencia y error cometido depende en gran medida de ciertos parámetros como el tamaño de la población, frecuencia de aplicación de los diferentes operadores, además de la elección de los diferentes operadores. Es asumido que la convergencia ha sido alcanzada cuando nada de la población inicial puede mejorar las soluciones. La suerte

ISBN: XXX-XXX-XX-XXXX-X

favorable es completa cuándo al menos un gen ha alcanzado convergencia(Coello et al., 2007).

Codificación.

La codificación se puede realizar de varias formas. La más utilizada es mediante una cadena de números binarios (1 o 0). Pero también se puede realizar la codificación mediante números enteros o incluso cadenas de palabras.

Las normas básicas a tener en cuenta para el diseño de una codificación óptima se pueden resumir en: Cada solución del problema tendrá su correspondiente cadena codificada, para cada cadena codificada producida mediante los operadores genéticos existirá su correspondiente solución decodificada, codificación y solución se corresponderán una a una, es decir no habrá redundancia, la codificación deberá permitir heredar de padres a hijos las características inherentes a los primeros (Fang, 1994).

Los algoritmos genéticos requieren que el conjunto se codifique en un cromosoma. Cada cromosoma tiene varios genes, que corresponden a ciertos parámetros del problema. Para poder trabajar con estos genes en el ordenador, es necesario codificarlos en una cadena, es decir, una ristra de símbolos (números o letras).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.

Problemas multiobjetivos.

Existen escenarios reales en los cuales la resolución del problema requiere optimizar objetivos los cuales pueden ser conflictivos entre ellos. Esta necesidad hace que las técnicas necesarias para la resolución de problemas multiobjetivos sean diferentes de las técnicas con un solo objetivo. Los problemas multimodales se caracterizan por tener varios óptimos locales y globales, por lo que los métodos tradicionales de optimización suelen caer atrapados en uno de los muchos óptimos locales obteniéndose bajos rendimientos. Además, dada las condiciones de trabajo puede ser necesario encontrar más de un óptimo global y de entre ellos seleccionar el que mejor se adapte a las condiciones reales del problema (Pérez et al., 2001, Coello et al., 2007, López, 2010, Back and Schwefel, 1993, Hajela and Ashley, 1981).

Operadores genéticos

Para el paso de una generación a la siguiente se aplican una serie de operadores genéticos. Los más empleados son los de selección, cruce, mutación y evaluación. En el caso de no trabajar con una población intermedia temporal también cobran relevancia los algoritmos de reemplazo.

Selección de padres.

Los algoritmos de selección serán los encargados de escoger qué individuos van a disponer de oportunidades de reproducirse y cuáles no. Puesto que se trata de imitar lo que ocurre en la naturaleza, se ha de otorgar un mayor número de oportunidades de reproducción a los individuos más aptos. No se debe, sin embargo, eliminar por completo las opciones de reproducción de los individuos menos aptos, pues en pocas generaciones la población se volvería homogénea.

ISBN: XXX-XXX-XX-XXXX-X

En cuanto a algoritmos de selección se refiere, estos pueden ser divididos en dos grandes grupos: probabilísticos y determinísticos. Ambos tipos de algoritmos basan su funcionamiento en permitir escoger una mayor cantidad de veces a los más aptos como se ha indicado anteriormente. Sin embargo, como su nombre indica, el primer tipo funciona con un importante componente basado en el azar. Es en este grupo donde se encuentran los algoritmos de selección por ruleta o por torneo estos tienen gran importancia por ser los que se utilizan con más frecuencia. El segundo grupo engloba una serie de algoritmos que, dado el ajuste conocido de cada individuo, permite asignar a cada uno el número de veces que será escogido para reproducirse. Esto puede evitar problemas de predominancia de ciertos individuos y cada uno de estos algoritmos presenta variaciones respecto al número de veces que se tomarán los mejores y peores.

El operador de selección se encarga de hacer duplicados de los mejores individuos que se tienen en la población y de eliminar los peores individuos de la misma, manteniendo un tamaño constante de la población. Se hace necesario mencionar que la selección no produce nuevos individuos en la población, sino que realiza copias de los mejores. Con la aplicación de este operador se crea un estanque de apareamiento el cual es utilizado por los operadores de cruzamiento y mutación, que se encargarán de crear los nuevos individuos. Existen varias formas de realizar la selección, muchas de las cuales primeramente identifican los mejores individuos de la población, hacen copias múltiples de estos individuos y finalmente eliminan los peores individuos para ser reemplazados con las copias de los mejores (Bremermann, 1962, Jenkins, 2002, Koza, 1992, Eiben and J.E.Smith, 2007, Coello et al., 2007).

Selección por ruleta (RWS)

Fue propuesta por De Jong, es posiblemente el más utilizado desde los orígenes de los Algoritmos Genéticos (Blickle&Thiele, 1995).Su funcionamiento consiste en simular una ruleta donde cada individuo x tiene asignado una sección de tamaño P(x) que representa la probabilidad que tiene ese elemento de ser escogido. Esta P(x) se calcula en función de su nivel de adaptación; es decir, del valor que nos devuelve la función de fitness, y que se normaliza entre 0 y 1 mediante la expresión. Una vez construida la ruleta se trata de obtener una serie de valores entre cero y uno que nos indicarán los elementos seleccionados como progenitores de la nueva generación.

Así pues, este método sigue una pauta elitista en tanto que, cuanto mejor adaptado esté un individuo, mayor será la sección de ruleta que se le asignara, y por tanto mayor probabilidad de ser escogido. Aun así no es un método puramente elitista. Presenta además el inconveniente de que el peor individuo puede ser seleccionado más de una vez(Jong, 1975, Fang, 1994, Greffenstette, 1986, Alander, 1992).

Selección aleatoria

Esta técnica selecciona aleatoriamente los padres de la nueva población. Presenta la ventaja de mantener la diversidad de la población y el inconveniente de hacer extremadamente lento el proceso de convergencia(Coello, 2014, Wang, 1991, Back and Schwefel, 1993,Hajela and Ashley, 1981).

Selección por torneo

ISBN: XXX-XXX-XX-XXXX-X

Esta es una de las más efectivas y sencillas de implementar, la idea principal de este método de selección consiste en escoger a los individuos genéticos en base a comparaciones directas entre sus genotipos. Existen dos versiones de selección mediante torneo, el torneo determinístico y el torneo probabilístico. En la versión determinística se selecciona al azar un número p de individuos (generalmente se escoge p=2). De entre los individuos seleccionados se selecciona el más apto para pasarlo a la siguiente generación.
La versión probabilística únicamente se diferencia en el paso de selección del ganador del torneo. En vez de escoger siempre el mejor se genera un número aleatorio si es mayor que un parámetro p (fijado para todo el proceso evolutivo) se escoge el individuo más apto y en caso contrario el menos apto. Cuando participan muchos individuos en cada torneo, la presión de selección es elevada y los peores individuos apenas tienen oportunidades de reproducción. Cuando por lo contrario el tamaño del torneo es reducido, la presión de selección disminuye y los peores individuos tienen más oportunidades de ser seleccionados (Thomas and Brown, 1977, Koza, 1992, Rechenberg, 1973).

Selección por muestreo universal estocástico (SUS)

También conocida como selección estocástica sin reemplazo. Fue definida originalmente por De Jong y estudiada por Baker con el fin de mejorar la mala distribución de los individuos de la población en función de los valores esperados obtenidos mediante el operador de selección por ruleta. En este caso se mapean los individuos en segmentos contiguos de una misma línea de manera que cada segmento es proporcional en tamaño a su aptitud, de modo similar a la selección por ruleta. Este operador presenta tres ventajas fundamentales. La primera es que reduce la alta variabilidad estocástica presente en la selección por ruleta. La segunda que reduce el tiempo de computación ya que el número de selecciones necesarias es menor. Por último y no menos importante, garantiza que si un individuo es suficientemente apto, será seleccionado incluso en varias ocasiones(Jong, 1975, Coello, 2014, Greffenstette and Baker, 1989).

Selección por muestreo determinístico

Se trata de una variación del operador por ruleta. En este caso cada individuo es seleccionado una cantidad de veces igual a la parte entera del valor esperado. A continuación se ordena la población según la parte no entera del valor esperado. Los individuos restantes en el proceso de selección se determinan en función del ordenamiento anterior (Greffenstette and Baker, 1989, Jong, 1975).

Selección por muestreo estocástico del resto con Reemplazo

Se trata de una variación del operador por muestreo determinístico. En este caso los individuos son también seleccionados en función de la parte entera del valor esperado. A continuación la parte no entera es utilizada para calcular la probabilidad de selección según el procedimiento de la ruleta de selección que es utilizada para seleccionar el resto de individuos(Greffenstette and Baker, 1989, Jong, 1975).

Selección por muestreo estocástico del resto sin Reemplazo

Al igual que el anterior se trata de una variación del operador por muestreo determinístico. En este caso los individuos son también seleccionados en función de la parte entera del valor esperado. A continuación la parte no entera es utilizada para lanzar monedas donde la probabilidad de que salga una cara u otra está ponderada por la parte no entera del valor

ISBN: XXX-XXX-XX-XXXX-X

esperado. De este modo, si por ejemplo, el número de copias esperadas de un individuo es de 1,7 este será seleccionado al menos una vez y tendrá una probabilidad del 70% de volver a ser seleccionado. El proceso es repetido hasta seleccionar el resto de individuos(Greffenstette and Baker, 1989, Jong, 1975,Coello et al., 2007).

Selección por grupos

También llamada selección por bloques, fue propuesta por Thierens y Goldberg. Cuando la presión selectiva es muy baja, los operadores de selección proporcionales a la aptitud retardan la convergencia del algoritmo. La implementación se realiza del siguiente modo: se divide la población en k grupos, asignándose una probabilidad de selección a cada grupo. La probabilidad de selección de un individuo de un grupo dado se obtiene dividiendo la probabilidad de selección del grupo por el número de individuos de dicho grupo(Thierens and Goldberg, 1994, Thomas and Brown, 1977,Coello et al., 2007).

Selección por rango

Esta fue concebida con el fin de superar los inconvenientes de la selección por ruleta. Este operador determina la probabilidad de selección de un individuo en función del rango que este posee dentro de la población en lugar de su aptitud. La determinación del rango se realiza del siguiente modo: el elemento con peor aptitud tiene un rango de 1 y el más apto recibe un rango μ. Seguidamente se seleccionan aleatoriamente dos individuos de la población y mediante un torneo definido por alguna regla relacionada con el rango se determina que individuo será el primer padre. A continuación se vuelven a seleccionar dos individuos y aplicando la misma regla se selecciona el segundo padre. El proceso se repite hasta generar todos los hijos(Bremermann, 1962, Jenkins, 2002, Koza, 1992, Eiben and J.E.Smith, 2007,Coello et al., 2007).

Selección del valor esperado.

Otra posible solución a los problemas derivados de súper individuos es la que ofrece el método de selección del valor esperado. Este sistema asigna un contador a cada individuo cuyo valor inicial es directamente proporcional a su función de fitness e inversamente proporcional a la media de dicha función en el instante en que aparece dicho individuo.

Este contador decrecerá en un valor comprendido entre 0,5 y 1 cada vez que el individuo sea seleccionado para ser progenitor. De esta forma el número de veces que un mismo individuo puede ser seleccionado está limitado y será descartado cuando el contador pase a ser negativo.

La elección de uno u otro método de selección definirán la estrategia de búsqueda del algoritmo genético. Utilizando métodos más elitistas la búsqueda se centra en los entornos próximos a las mejores soluciones. Utilizando métodos dónde la presión de la selección sea menor se está dejando abiertas las puertas a la exploración de zonas desconocidas que pueden reportar mejores soluciones o simplemente tiempo de cálculo perdido.

Reproducción o cruce (crossover).

Una vez seleccionados los individuos, éstos son recombinados para producir la descendencia que se insertará en la siguiente generación. Este es el principal operador de búsqueda en los algoritmos genéticos y tiene como principio la reproducción sexual.

Los diferentes métodos de cruce podrán operar de dos formas diferentes. De optarse por una estrategia destructiva los descendientes se insertarán en la población temporal aunque

ISBN: XXX-XXX-XX-XXXX-X

sus padres tengan mejor ajuste (trabajando con una única población esta comparación se realizará con los individuos a reemplazar). Por el contrario, utilizando una estrategia no destructiva la descendencia pasará a la siguiente generación únicamente si supera la bondad del ajuste de los padres (o de los individuos a reemplazar).

La idea principal del cruce se basa en que, si se toman dos individuos correctamente adaptados al medio y se obtiene una descendencia que comparta genes de ambos, existe la posibilidad de que los genes heredados sean precisamente los causantes de la bondad de los padres. Al compartir las características buenas de dos individuos, la descendencia, o al menos parte de ella, debería tener una bondad mayor que cada uno de los padres por separado. Si el cruce no agrupa las mejores características en uno de los hijos y la descendencia tiene un peor ajuste que los padres, no significa que se esté dando un paso atrás. Optando por una estrategia de cruce no destructiva se garantiza que pasen a la siguiente generación los mejores individuos. Si, aún con un ajuste peor, se opta por insertar a la descendencia, y puesto que los genes de los padres continuarán en la población aunque dispersos y posiblemente levemente modificados por la mutación, en posteriores cruces se podrán volver a obtener estos padres. La combinación de la información genética de los padres permite realizar búsquedas en otros puntos del espacio de soluciones. Cuanto más diferentes sean los padres, tanto mayor seria la influencia de este operador en el funcionamiento del algoritmo.

Por lo tanto, la eficiencia de la búsqueda dependería en gran medida de la presión selectiva y la diversidad de la población. Aunque generalmente intervienen dos padres y se generan dos hijos, esta no es la forma exclusiva de cruce. El cruce podría extenderse a un caso más general donde intervienen varios padres o se genera un solo individuo(Díaz, 2006, Mitchell, 1996, Grierson and Pak, 1993, Rajeev and Krishnamoorthy, 1997, Back and Schwefel, 1993, Darwin, 1866, Jong, 1975, Coello et al., 2007).

Probabilidad de cruce.

Una vez elegidos a los progenitores, es la probabilidad de cruce la que decidirá si estos intercambiarán su material genético. Indica la frecuencia con la que se producen cruces entre los cromosomas padres 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 (Greffenstette, 1986, Jenkins, 2002, Lawo and Thierauf, 1982, Darwin, 1866, Fang, 1994, Coello et al., 2007).

Tipos de cruce.

Existen multitud de algoritmos de cruce. Sin embargo los más empleados son:

Cruce de 1 punto

Es la más sencilla de las técnicas de cruce. Una vez seleccionados dos individuos se cortan sus cromosomas por un punto seleccionado aleatoriamente para generar dos segmentos diferenciados en cada uno de ellos: la cabeza y la cola. Se intercambian las colas entre los dos individuos para generar los nuevos descendientes. De esta manera ambos descendientes heredan información genética de los padres (Greffenstette, 1986, Jenkins, 2002, Lawo and Thierauf, 1982, Darwin, 1866, Fang, 1994, Jong, 1975, Coello et al., 2007).

ISBN: XXX-XXX-XX-XXXX-X

Cruce de 2 puntos

Se trata de una generalización del cruce de 1 punto. En vez de cortar por un único punto los cromosomas de los padres, como en el caso anterior, se realizan dos cortes. Deberá tenerse en cuenta que ninguno de estos puntos de corte coincida con el extremo de los cromosomas para garantizar que se originen tres segmentos. Para generar la descendencia se escoge el segmento central de uno de los padres y los segmentos laterales del otro padre. Generalmente, es habitual referirse a este tipo de cruce con las siglas DPX (Double Point Crossover) (Greffenstette, 1986, Jenkins, 2002, Lawo and Thierauf, 1982, Darwin, 1866, Fang, 1994, Jong, 1975, Coello et al., 2007).

Generalizando, se pueden añadir más puntos de cruce dando lugar a algoritmos de cruce multipunto. Sin embargo De Jong investigó el comportamiento del operador de cruce basado en múltiples puntos, concluyendo que el cruce basado en dos puntos, representaba una sustancial mejora con respecto al cruce de un solo punto pero mientras que se seguía añadiendo más puntos de cruce no beneficiaba el comportamiento del algoritmo al contrario reduce el rendimiento del Algoritmo Genético. La ventaja de tener más de un punto de cruce radica en que el espacio de búsqueda puede ser explorado más fácilmente y con mayor intensidad, siendo la principal desventaja el hecho de aumentar la probabilidad de ruptura de buenos esquemas (Jong, 1975, Coello et al., 2007).

Cruce uniforme

El cruce uniforme es una técnica completamente diferente de las vistas hasta el momento. Cada gen de la descendencia tiene la misma probabilidad desde pertenecer a uno u otro padre. Aunque se puede implementar de muy diversas formas, la técnica implica la generación de una máscara de cruce con valores binarios. Si en una de las posiciones de la máscara hay un 1, el gen situado en esa posición en uno de los descendientes se copia del primer padre. Si por el contrario hay un 0 el gen se copia del segundo padre. Para producir el segundo descendiente se intercambian los papeles de los padres, o bien se intercambia la interpretación de los unos y los ceros de la máscara de cruce. La máscara de cruce puede no permanecer fija durante todo el proceso evolutivo. Se genera de manera aleatoria para cada cruce. Se suele referir a este tipo de cruce con las siglas UPX (Uniform Point Crossover).

Mutación.

En la evolución, una mutación es un suceso bastante poco común. Estas 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 el nuevo individuo a partir de sus padres (normalmente se hace de forma simultánea al cruce). 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, esto ocurre en el caso de una codificación binaria, la mutación consiste simplemente en la inversión del gen mutado que corresponde con un bit. 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. En el caso de una codificación numérica, la mutación podría consistir en sustituir un número por otro o intercambiar un número que está en otra posición del cromosoma. En el caso de codificación por valor directo en el que por ejemplo se usen

ISBN: XXX-XXX-XX-XXXX-X

números reales, la mutación puede consistir simplemente en modificar el valor en unos decimales. 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(Darwin, 1866, Koza, 1992, Coello, 2014, Thomas, 2009, Rechenberg, 1965).

Los algoritmos genéticos son capaces de explorar y explotar el espacio de soluciones. Por explotar se entiende realizar una búsqueda exhaustiva en una zona restringida del espacio. Esta tarea se realiza mediante el operador cruce, que actúa como un intensificador de la búsqueda en la zona del espacio de soluciones actualmente ocupada por la población. El efecto exploratorio, por el que se prospectan zonas del espacio alejadas de las soluciones presentes en la población, es asumido por el operador mutación(Thomas, 2009, Rechenberg, 1965, Back and Schwefel, 1993).

La probabilidad de mutación es muy baja, generalmente menor al 1 %. Esto se debe sobre todo a que los individuos suelen tener un ajuste menor después de mutados. Sin embargo a modo de conclusión se realizan mutaciones para garantizar que ningún punto del espacio de búsqueda tenga una probabilidad nula de ser examinado (Davis, 1991, Thomas, 2009, Rechenberg, 1965,Back and Schwefel, 1993).

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 (Davis, 1989, Thomas, 2009, Rechenberg, 1965, Coello et al., 2007).

Según lo expuesto anteriormente se observa que la elección de una adecuada probabilidad de mutación es un elemento crucial. Si la probabilidad de mutación es muy elevada, el algoritmo genético se convierte en una búsqueda aleatoria; mientras que si se elige una probabilidad de mutación muy pequeña, entonces la pérdida de diversidad genética producida cuando los individuos de la población están convergiendo, puede llevar al algoritmo a un subóptimo. La determinación de la probabilidad de mutación es, de hecho, mucho más crucial que la probabilidad de cruce. La búsqueda del valor óptimo para la probabilidad de mutación, es una cuestión que ha sido motivo de muchos trabajos de investigación. Así, una de las recomendaciones es la utilización de una probabilidad de mutación de 1/l (l es la longitud de la cadena)(Jong, 1975, Back and Schwefel, 1993, Coello et al., 2007).

La mutación más usual es el reemplazo aleatorio. Este consiste en variar aleatoriamente un gen de un cromosoma. Si se trabaja con codificaciones binarias, consistirá simplemente en negar un bit.

También es posible realizar la mutación intercambiando los valores de dos alelos del cromosoma. Con otro tipo de codificaciones no binarias existen otras opciones:

- Aumentar o disminuir a un gen una pequeña cantidad generada aleatoriamente.
- Multiplicar un gen por un valor aleatorio próximo a 1.

ISBN: XXX-XXX-XX-XXXX-X

Aunque no es lo más común, existen implementaciones de Algoritmos Genéticos en las que no todos los individuos tienen los cromosomas de la misma longitud. Esto implica que no todos ellos codifican el mismo conjunto de variables. En este caso existen mutaciones adicionales como puede ser añadir un nuevo gen o eliminar uno ya existente (Bremermann, 1962, Jenkins, 2002, Koza, 1992, Eiben and J.E.Smith, 2007,Coello et al., 2007).

Algunos ejemplos de operadores de mutación más usados:

Mutación estándar. Es el operador por excelencia utilizado en la codificación binaria. La probabilidad de aplicación es muy baja, por no incluir demasiada diversidad en la búsqueda y perder de esta forma la dirección de la misma. Según los estudios realizados por Holland, la probabilidad aceptable sería la comprendida entre 0.1 y 1%. Esta probabilidad se aplicará bit a bit, a diferencia de los operadores cruce que se refieren a cada pareja o incluso a otros operadores mutación, generalmente los aplicados sobre codificaciones no binarias, donde se aplicará sobre cada cadena o individuo(Holland, 1962, Coello, 2014).

Mutación no uniforme. Desarrollado para codificaciones reales. El operador mutación estándar actúa de diferente forma en la codificación binaria que en la codificación real, de tal forma que el operador así diseñado actúa de una forma mucho más aleatoria para la codificación real que para la binaria, donde cambiar un bit aleatoriamente no implica producir un valor totalmente aleatorio del dominio. Para evitar este efecto se diseñó la mutación no uniforme (Mitchell, 1996, Holland, 1962, Coello, 2014).

Mutación por permutación de un subconjunto fue desarrollada por Davis para la codificación permutacional, pero que es directamente aplicable también a la permutacional con repetición. Se seleccionan dos posiciones de la cadena correspondiente al individuo que tiene que mutar, y los valores comprendidos entre ellas se permutan de forma aleatoria, dejando el resto del cromosoma igual(Davis, 1991, Holland, 1962, Coello, 2014).

Evaluación.

Para el correcto funcionamiento de un Algoritmo Genético se debe de poseer un método que indique si los individuos de la población representan o no buenas soluciones al problema planteado. Por lo tanto, para cada tipo de problema que se desee resolver deberá derivarse un nuevo método, al igual que ocurrirá con la propia codificación de los individuos.

La aproximación más común consiste en crear explícitamente una medida de ajuste para cada individuo de la población. A cada uno de los individuos se le asigna un valor de ajuste escalar por medio de un procedimiento de evaluación bien definido. Tal y como se ha comentado, este procedimiento de evaluación será específico del dominio del problema en el que se aplica el Algoritmo Genético. También puede calcularse el ajuste mediante una manera ‘coevolutiva’. Por ejemplo, el ajuste de una estrategia de juego se determina aplicando esa estrategia contra la población entera (o en su defecto una muestra) de estrategias de oposición(Greffenstette and Baker, 1989, Coello et al., 2007, Goldberg, 1989, Jong, 1975).

Para conocer si el algoritmo genético está funcionando bien el modo más simple sería esperar a ver los resultados finales y comprobar si los valores son aceptables. Según De Jong, los métodos que se exponen a continuación permiten evaluar la progresión que está siguiendo el AG, de manera que se pueda valorar si está funcionando correctamente.

ISBN: XXX-XXX-XX-XXXX-X

Evaluación On-Line
Utilizando este método a lo largo de las diferentes generaciones, se puede observar la evolución media de todos los individuos hasta la generación actual.
Evaluación Off-Line:
En este método se puede observar cómo va evolucionando el valor óptimo alcanzado en cada generación a lo largo de T generaciones. Donde fmaxfitness (t) representa el valor más alto de fitness alcanzado por un individuo dentro de la generación t(Jong, 1975).

Sustitución o Reemplazo.

Existen dos tipos diferentes de algoritmos genéticos según sea el proceso de la reproducción y que afectará a la forma de aplicar la sustitución.
- Algoritmos genético generacionales. Donde de la población inicial formada por (N) individuos se obtienen (N) padres que son recombinados mediante los operadores genéticos apropiados para dar lugar a (N) hijos. Después, estos sustituirán directamente a la población inicial siendo así la población de partida de la siguiente generación cerrando el ciclo.
- Algoritmos genéticos de creación continúa. En esta técnica se seleccionan dos individuos para obtener dos hijos, siendo incorporados a la población mediante la sustitución de uno o dos individuos. La sustitución puede ser:
1. Totalmente aleatoria. Todos tienen la misma probabilidad.
2. Seleccionando directamente a los peor o peores individuos.
3. Mediante técnicas de clasificación u ordenación (Goldberg, 1989, Jong, 1975, Thomas and Brown, 1977, Alander, 1992,Coello, 2014).

Una modificación destacable de esta técnica se denomina creación continua sin duplicados desarrollada para el programa GENITOR, cuya principal característica es que evitan trabajar con individuos duplicados al descartar hijos ya presentes en la población. El principal beneficio de esta técnica frente a las convencionales, es el uso más eficiente de la población al evaluar sólo individuos diferentes. El inconveniente más importante es el aumento de tiempo del proceso por la identificación y eliminación de copias. (Syswerda, 1989, Sarma and Jong, 1996)
La principal diferencia entre los algoritmos genéticos generacionales y los de creación continua es que cuando se crea un buen individuo, éste estará disponible para el proceso inmediatamente, mientras que en el reemplazo generacional hay que esperar a la siguiente generación. En los estudios que llevó a cabo De Jong, demostró que el modelo generacional, era mejor para los problemas de optimización.
Al utilizar los algoritmos genéticos generacionales se emplea habitualmente el proceso elitista, este método aporta que cuando se ha formado ya la población de hijos que será la partida de la siguiente generación se elimina al peor individuo y en su lugar pasa directamente al mejor individuo de la población inicial de la generación anterior. De esta forma, el proceso tendrá siempre una copia, al menos, de la mejor solución encontrada en cada momento, por si por el efecto de la selección y de los operadores genéticos, ésta se perdiera (Grierson and Pak, 1993, Caballero, 2012, Goldberg, 1989, Jong, 1975).
Las diferencias entre las estrategias adoptadas para los Algoritmos genéticos de creación continúa y los Algoritmos Genéticos con brecha generacional dan lugar a los diferentes operadores de reemplazo:

ISBN: XXX-XXX-XX-XXXX-X

- Reemplazo de los menos aptos: Fue definido por De Jong y Sarma dentro de un modelo de Algoritmo Genético incremental denominado GENITOR. Con este operador los hijos engendrados reemplazan a los individuos menos aptos de la población(Sarma and Jong, 1996).
- Reemplazo aleatorio: Definido por Syswerda con este operador los hijos engendrados reemplazan de forma aleatoria a los individuos de la población(Syswerda, 1989, Syswerda, 1991).
- Torneo a muerte: Definido por Smith con este operador se selecciona aleatoriamente a un conjunto de individuos de la población, donde el menos apto es sustituido por un hijo (Smith, 1980).
- Reemplazo del individuo más viejo: Con este operador se reemplaza al individuo más viejo. Se corre el riesgo de eliminar al más apto si no se incorpora ningún operador de elitismo (Thierens and Goldberg, 1994).
- Selección conservativa: Definido por Smith combina la estrategia de reemplazo del más viejo con un torneo de selección binaria. Este operador realiza un torneo entre el individuo más viejo y otro seleccionado aleatoriamente dentro de la población. Si el más viejo es más apto sobrevive y el otro es sustituido por un hijo. En caso contrario es el viejo el reemplazado. Esta estrategia tiene la ventaja de no reemplazar nunca al individuo más apto si es también el más viejo(Baker, 1987, Syswerda, 1991).
- Reemplazo de los progenitores: Este operador mediante una estrategia de supervivencia predefinida decide si el hijo reemplaza a alguno de los padres o bien ambos progenitores sobreviven(Jong, 1975).
- Elitismo: El operador elitista tiene por objeto evitar la eliminación del individuo o grupo de individuos más aptos de la población. La eliminación del individuo más apto puede provocar una pérdida significativa de explotación dificultando la convergencia del algoritmo. Casi todos los operadores de reemplazo actuales y algunos de cruce incorporan estrategias elitistas(Wang, 1991, Back and Schwefel, 1993, Baker, 1987).

Ventajas, desventajas y limitaciones de los algoritmos genéticos.

Ventajas:

- No necesitan conocimientos específicos del problema a resolver.
- Operan de forma simultánea con varias soluciones y no de forma secuencial como las técnicas tradicionales.
- Son fáciles de ejecutar en modernas arquitecturas masivamente paralelas.
- Usan operadores probabilísticos, en vez de los típicos operadores determinístico de las otras operaciones.
- Trabajan con un código del conjunto de parámetros, no con el conjunto mismo. Por trabajar a nivel de código, y no con las funciones y sus variables de control, como los otros métodos, es más difícil que la solución converja a un mínimo o máximo local.

ISBN: XXX-XXX-XX-XXXX-X

- Buscan una población de puntos, no un único punto. Manteniendo una población de puntos muestrales bien adaptados, se reduce la probabilidad de caer en una solución falsa.
- Emplean la función objetivo, no necesitan derivadas ni otra información complementaria, tan difícil a veces de conseguir. De este modo ganan en eficiencia y en generalidad.
- Cuando se usan para problemas de optimización resultan menos afectados por los máximos o mínimos locales que las técnicas tradicionales.
- Se valen de reglas de transición estocástica, no deterministas. Los Algoritmos Genéticos se valen de operadores aleatorios para guiar la búsqueda de los mejores puntos.
- Pueden tardar mucho en converger, o no converger en absoluto, dependiendo en gran medida de los parámetros que se utilicen: tamaño de la población, número de generaciones y porcentajes de los operadores genéticos (Coello et al., 2007, Goldberg, 1989, Holland, 1975, Jong, 1975, Jenkins, 2002, Thierens and Goldberg, 1994, Alander, 1992, Wang and Arora, 2004, Fraser, 1958, Thomas, 2009, Fogel, 1967).

Desventajas:

- Pueden tardar mucho en converger o no converger en absoluto, dependiendo en cierta medida de los parámetros que utilicen tamaño de la población, numero de generaciones, etc.
- Pueden converger prematuramente debido a una serie de problemas de diversa índole(Coello et al., 2007, Goldberg, 1989, Holland, 1975, Jong, 1975, Jenkins, 2002, Thierens and Goldberg, 1994, Alander, 1992, Wang and Arora, 2004, Fraser, 1958).

Limitaciones

Como se ha visto a lo largo de este trabajo de investigación, los Algoritmos Genéticos son una técnica robusta que pueden tratar con éxito una gran variedad de problemas provenientes de varias áreas, incluyendo métodos en los que encuentran dificultades. Hay que tener en cuenta que el uso de los Algoritmos Genéticos no garantiza que encuentre una solución óptima del problema, existe evidencia de que encuentra soluciones de un nivel aceptable, en un tiempo competitivo con el resto de las demás técnicas de optimización, incluso pueden existir técnicas que sean más rápidas y eficaces que el mismo Algoritmo Genético, pese a todas esas limitaciones; este tipo de algoritmos es una gran herramienta que siguen siendo la opción más aceptable al resolver problemas(Coello et al., 2007, Goldberg, 1989, Holland, 1975, Jong, 1975, Jenkins, 2002, Thierens and Goldberg, 1994, Alander, 1992, Wang and Arora, 2004, Fraser, 1958).

Conclusiones:

Los algoritmos genéticos junto con las estrategias y la programación evolutivas constituyen un nuevo enfoque a la solución de numerosos problemas de optimización y búsqueda dentro de lo que se puede destacar su aplicación en diversas ramas de la ciencia. Los algoritmos genéticos se basan en el cruce como el operador de búsqueda más importante. A

ISBN: XXX-XXX-XX-XXXX-X

diferencia de las estrategias evolutivas, el operador de mutación se aplica con una probabilidad muy pequeña, y no es considerado un operador fundamental. Se usa la selección probabilística y, generalmente, se emplea la codificación de los individuos mediante cadenas binarias, aunque también se pueden emplear otras codificaciones tales como las basadas en números reales. Los algoritmos genéticos suelen ser eficaces con funciones no derivables, y hay que tener en cuenta que si existen muchos máximos/mínimos locales se necesitan de más iteraciones para asegurar el global, y que si tiene muchos puntos cercanos al óptimo, no podemos asegurar que encontraremos dicho óptimo, pero sí al menos una buena solución. El tamaño de población y el número de generaciones son parámetros que intervienen directamente sobre el funcionamiento de los algoritmos genéticos si la medida de ambos es insuficiente, se realiza una búsqueda de soluciones escasa y poco óptima. Por otro lado si estos parámetros son grandes, el algoritmo genético será excesivamente lento. De modo que se debe tener en cuenta que hay un límite a partir del cual es ineficiente elevar el tamaño de la población y la cantidad de generaciones puesto que no se consigue una mejoría en la solución del problema.

ISBN: XXX-XXX-XX-XXXX-X

Bibliografía

ADELI, H. & CHENG, N.-T. 1993. Integrated genetic algorithm for optimization of space structures. Journal of Aerospace Engineering, 6(4):315–328.

AI, S. & WANG, Y. 2011. Application of improved genetic algorithms in structural optimization design. In M. Zhu, editor, Information and Management Engineering, volume 236 of Communications in Computer and Information Science, pages 480–487. .

ALANDER, J. T. 1992. On optimal population size of genetic algorithms. In Computer Systems and Software Engineering, 6th Annual European Computer Conference.

ALBA, E. & TOMASSINI, M. 2002. Parallelism and evolutionary algorithms.

AMIRJANOV, ADIL, SOBOLEV & KONSTANTIN 2005. Optimal proportioning of concrete aggregates using a self-adaptive genetic algorithm. Computers and Concrete, vol. 2, no 5, p. 411-421.

BACK, T., HO, F. & SCHWEFEL, H. P. 1991. A survey of evolution strategies. In Morgan Kaufmann, editor, Proceedings of the Fourth International Conference on Genetic Algorithms, pages 2-9, San Mateo, CA, .

BACK, T. & SCHWEFEL, H. P. 1993. An overview of evolutionary algorithms for parameter optimization. Evolutionary Computation.

BAKER, J. E. 1987. Reducing bias and inefficiency in the selection algorithm. In Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application Hillsdale, NJ, USA, 1987. L. Erlbaum Associates Inc. ISBN 0-8058-0158-8, pages 14–21.

BIETHAHN, J. & NISSEN, V. 1995. Evolutionary algorithms in management applications. Springer. New York.

BOTELLO, S., MARROQUÍN, J. L., OÑATE, E. & HOREBEEK, J. V. 1999. Solving structural optimization problems with genetic algorithms and simulated annealing. International Journal for Numerical Methods in Engineering, 45(8):1069–1084.

BREMERMANN, H. J. 1962. Optimization through evolution and recombination. In: Self-Organizing systems 1962, edited M.C. Yovitts et al., Spartan Books, Washington, D.C. pp. 93–106.

CABALLERO, S. S. 2012. OPTIMIZACION ESTRUCTURAL Y TOPOLOGICA DE ESTRUCTURAS MORFOLOGICAMENTE NO DEFINIDAS MEDIANTE ALGORITMOS GENETICOS. Doctoral, UNIVERSITAT POLITECNICA DE VALENCIA.

CALDERON, J. A. O. 2005. USO DE ALGORITMOS GENÉTICOS PARA FACILITAR EL MANEJO DE LAS VARIABLES Y SU RELACIÓN CON EL VALOR PRESENTE NETO, EN LA CONSTRUCCIÓN DE UN PARQUE URBANO Diplomado, UNIVERSIDAD JAVERIANA BOGOTA.

COELLO, C. A. C. 1999. Treating constraints as objectives for single-objective evolutionary optimization. Engineering Optimization, 32:275–308.

COELLO, C. A. C. 2014. Introducción a la Computación Evolutiva. CINVESTAV-IPN,.

ISBN: XXX-XXX-XX-XXXX-X

COELLO, C. A. C., LAMONT, G. B. & VELDHUIZEN, D. A. V. 2007. Evolutionary Algorithms for Solving Multi-Objective Problems.
CHAU, K. W. & ALBERMANI, F. 2003. Knowledge-based system on optimum design of liquid retaining structures with genetic algorithms. Journal of structural engineering, vol. 129, no 10, p. 1312-1321.
DARWIN, C. 1866. The origin of species by means of natural selection, or the preservation of favoured races in the struggle for life. J. Murray, London.
DAVIS, L. 1989. Adapting operator probabilities in genetic algorithms. Proceedings of the third international conference on Genetic Algorithms. J. David Schaffer (Ed.). Kaufmann. San Mateo. Pag. 375-378. .
DAVIS, L. 1991. Handbook of genetics algorithms. Van Nostrand. New York.
DÍAZ, B. D. 2006. Diseño e Implementación de Algoritmos Genéticos Celulares para Problemas Complejos. Doctorado, UNIVERSIDAD DE MÁLAGA.
EIBEN, A. E. & J.E.SMITH 2007. Introduction to Evolutionary Computing Springer, 2nd printing, ISBN 3540401849.
FANG, H. 1994. Genetic Algorithms in Timetabling and Scheduling. Doctoral dissertation, University of Edinburg.
FOGEL, D. B. 1967. Artificial Intelligence Through Simulated Evolution. Wiley-IEEE Press. New York: Wiley Publishing.
FRASER, A. 1958. Monte carlo analyses of genetic models. Nature, 181(4603):208–209, Jan.
FRASER, A. S. 1957. Simulation of genetic systems by automatic digital computers II: E®ects of linkage on rates under selection. Australian Journal of Biological Sciences.
GERO, M. B. P., GARCÍA, A. B. & DÍAZ, J. J. D. C. 2005. A modified elitist genetic algorithm applied to the design optimization of complex steel structures. Journal of Constructional Steel Research, 61(2):265–280. ISSN 0143-974X.
GERO, M. B. P., GARCÍA, A. B. & DÍAZ, J. J. D. C. 2006. Design optimization of 3d steel structures: Genetic algorithms vs. classical techniques. Journal of Constructional Steel Research, ISSN 0143-974X. 62(12):1303–1309.
GOLDBERG, D. E. 1989. Genetic Algorithms in Search, Optimization and Machine Learning.
GREFFENSTETTE, J. J. 1986. Optimization of control parameters for genetic algorithms. Systems, Man and Cybernetics, IEEE Transactions on, 122–128.
GREFFENSTETTE, J. J. & BAKER, J. E. 1989. How genetic algorithms work: A critical look at implicit parallelism. In Proceedings of the 3rd International Conference on Genetic Algorithms, Morgan Kaufmann Publishers Inc. ISBN 1-55860-066-3, pages 20–27, San Francisco, CA, USA.
GRIERSON, D. & PAK, W. 1993. Optimal sizing, geometrical and topological design using a genetic algorithm. Structural and Multidisciplinary Optimization, 6 (3):151–159.
HAJELA, P. & ASHLEY, H. 1981. Hybrid optimization of truss structures with strength and buckling constraints. Tucson, AZ, USA. Conference of International Symposium on Optimum Structural Design. 11th ONR Naval Structural Mechanics Symposium (US Office of Naval Research).; Conference Code: 499. pages 3. 11–3. 18.

ISBN: XXX-XXX-XX-XXXX-X

HAJELA, P. & LEE, E. 1993. Genetic algorithms in topological design of grillage structures. Proc., IUTAM Symp. on Discrete Structural Systems, IUTAm, Zakopane, Poland.
HAJELA, P. & LIN, C. Y. 1992. Genetic search strategies in multicriterion optimal design. Structural and Multidisciplinary Optimization, 4(2):99–107.
HOLLAND, J. H. 1962. Outline for a logical theory of adaptive systems, JACM 9(3):297–314. DOI 10.1145/321127.321128.
HOLLAND, J. H. 1975. Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, MI.
JENKINS, W. 1991. Towards structural optimization via the genetic algorithm. Computers & Structures, 40(5):1321–1327.
JENKINS, W. 1997. On the application of natural algorithms to structural design optimization. Engineering Structures, ISSN 0141-0296.19(4):302–308.
JENKINS, W. 2002. A decimal-coded evolutionary algorithm for constrained optimization. Computers and Structures, ISSN 00457949, 80(5-6):471–480.
JENKINS, W. M. 1992. Plane Frame Optimum Design Environment Based on Genetic Algorithm. Journal of Structural Engineering. 118(11): 3103-3113.
JONG, K. A. D. 1975. An analysis of the behavior of a class of genetic adaptive systems, University of Michigan, Ann Arbor, MI, USA.
KOUMOUSIS, V. K. & GEORGIOU, P. G. 1994. Genetic algorithms in discrete optimization of steel truss roofs. Journal of Computing in Civil Engineering, 8(3): 309–325.
KOZA, J. R. 1992. Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press. ISBN 0-262-11170-5.
LAWO, M. & THIERAUF, G. 1982. Optimal design for dynamic stochastic loading: A solution by random search. optimization in structural design. university of siegen. Bibl. Inst. Mannheim, 346-352.
LÓPEZ, F. C. 2010. Optimización global con algoritmos genéticos. Diplomado, Universidad Politécnica de Catalunya.
MARIÑELARENA, D. A. 2013. Diseño de algoritmos genéticos para la detección de daños en estructuras, Publicación Técnica No. 386 Instituto Mexicano de Tansporte, ISSN 0188-7297
Sanfandila, Qro.
MITCHELL, M. 1996. An Introduction to Genetic Algorithms MIT Press. ISBN 0262133164.
NEHDI, M., CHABIB, H. E. & SAID, A. 2011. Genetic algorithm model for shear capacity of RC beams reinforced with externally bonded FRP. Materials and Structures 44:1249–1258.
RAFIQ & MOHAMMAD, Y. 1998. Genetic algorithms in optimal design and detailing of reinforced concrete biaxial columns supported by a declarative approach for capacity checking. Computers&structures, 1998, vol. 69, no 4, p. 443-457.
RAJAN, S. D. 1995. Sizing, shape, and topology design optimization of trusses using genetic algorithm. Journal of Structural Engineering, 121(10):1480–1487.

ISBN: XXX-XXX-XX-XXXX-X

RAJEEV, S. & KRISHNAMOORTHY, C. S. 1992. Discrete Optimization of Structures Using Genetic Algorithms. Journal of Structural Engineering. 118(5): 1233-1250.
RAJEEV, S. & KRISHNAMOORTHY, C. S. 1997. Genetic algorithms-based methodologies for design optimization of trusses. Journal of Structural Engineering, 123 (3):350–358, 1997.
RECHENBERG, I. 1965. Cybernetic solution path of an experimental problem. Stuttgart-Bad Cannstatt : Frommann-Holzboog.
RECHENBERG, I. 1973. Evolutionsstrategie : Optimierung technischer Systemenach Prinzipien der biologischen Evolution. Stuttgart-Bad Cannstatt : Frommann-Holzboog.
SARMA, J. & JONG, K. A. D. 1996. An analysis of the e®ect of the neighborhood size and shape on local selection algorithms. In H.M. Voigt, W. Ebeling, I. Rechenberg, and H.P. Schwefel, editors, Proc. of the International Conference on Parallel Problem Solving from Nature IV (PPSN-IV), volume 1141 of Lecture Notes in Computer Science (LNCS), pages 236-244. Springer-Verlag, Heidelberg.
SCHWEFEL, H. P. 1977. Numerische Optimierunguon Computer-Modellenmittels der Ezdutionsstrategie, Volume 26 of Interdisciplinary systems research. Basel Birkhauser.
SMITH, S. F. 1980a. A learning system based on genetic adaptive algorithms, University of Pittsburgh, PA, USA.
SMITH, S. F. 1980b. A learning system based on genetic adaptive algorithms. University of Pittsburgh, Pittsburgh, PA, USA.
SYSWERDA, G. 1989. Uniform crossover in genetic algorithms ".Proceedings of the third international conference on Genetic Algorithms. Schaffer, J. D. (Ed.). Kaufmann. San Mateo. Pag. 2-9.
SYSWERDA, G. 1991. A study of reproduction in generational and steady-state genetic algorithms". Foundations of Genetic Algorithms 1. Rawlins G.(Ed). Kaufmann. San Mateo. Pag. 94-101.
THIERENS, D. & GOLDBERG, D. E. 1994. Convergence models of genetic algorithm selection schemes. In Proceedings of the International Conference on Evolutionary Computation. The Third Conference on Parallel Problem Solving from Nature: Parallel Problem Solving from Nature, London, UK. Springer-Verlag. ISBN 3-540-58484-6. pages 119–129
THOMAS, H. & BROWN, D. 1977. Optimum least-cost design of a truss roof system. Computers and Structures, 7(1):13–22. ISSN 00457949.
THOMAS, W. 2009. Global optimization algorithms - theory and application. PDF.
VALDIVIA, R. A. D. G. 2014. OPTIMIZACIÓN DE ESTRUCTURAS RETICULADAS PLANAS DE MADERA MEDIANTE ALGORITMOS GENÉTICOS. Universidad Austral de Chile.
WANG, Q. & ARORA, J. S. 2004. Alternate formulations for structural optimization. volume 2, pages 1413–1423, Palm Springs, CA, .
WANG, Q. J. 1991. The genetic algorithm and its application to calibrating conceptual rainfall-runoff models. Water Resources Research.

ISBN: XXX-XXX-XX-XXXX-X