Introducción a la Computación Evolutiva

Dr. Carlos A. Coello Coello

Departamento de Computación

CINVESTAV-IPN

Av. IPN No. 2508

Col. San Pedro Zacatenco

México, D.F. 07300

email: ccoello@cs.cinvestav.mx

http: //delta.cs.cinvestav.mx/~ccoello

Representaciones Adaptativas

Codificación Delta

La idea de esta propuesta [Matthias & Whitley 1994] es cambiar dinámicamente la representación de un problema. Nótese, sin embargo, que no intenta "aprender" cuál es la mejor representación del espacio de búsqueda, sino más bien se cambia la representación de manera periódica para evitar los sesgos asociados con una representación determinada del problema.

Representaciones Adaptativas

El algoritmo de la codificación delta empieza con la ejecución inicial de un algoritmo genético usando cadenas binarias. Una vez que la diversidad de la población ha sido explotada adecuadamente, se almacena la mejor solución bajo el nombre de "solución temporal". Se reinicializa entonces el AG con una nueva población aleatoria. En esta ocasión, sin embargo, las variables se decodifican de tal forma que representen una distancia o *valor delta* ($\pm\delta$) con respecto a la solución temporal.

Representaciones Adaptativas

El valor de δ se combina con la solución temporal de manera que los parámetros resultantes se evalúen usando la misma función de aptitud.

De esta manera, la codificación delta produce un nuevo mapeo del espacio de búsqueda a cada iteración. Esto permite explorar otras representaciones del problema que podrían "facilitar" la búsqueda.

Representaciones Adaptativas

Ejemplo de codificación binaria usando códigos delta.

Parámetros numéricos	0	1	2	3	4	5	6	7
Codificación binaria	000	001	010	011	100	101	110	111
Cambios numéricos	0	1	2	3	-3	-2	-1	-0
Codificación delta	000	001	010	011	111	110	101	100

Tamaño de Población Adaptativo

- Smith [1993] propuso un algoritmo que ajusta su tamaño de población con respecto a la probabilidad del error de selección.
- Hinterding et al. [1996] experimentaron con un AG adaptativo, el cual consistía de 3 subpoblaciones. A intervalos regulares, los tamaños de estas subpoblaciones se ajustaban con base en el estado actual de la búsqueda.

Tamaño de Población Adaptativo

- Schlierkamp-Voosen and Mühlenbein [1994] usan un esquema de competencia que cambia los tamaños de las subpoblaciones, manteniendo fijo el número total de individuos.
- Arabas et al. [1994] propusieron un algoritmo genético con tamaño de población variable (GAVaPS), que usa el concepto de "edad" de un cromosoma, el cual es equivalente al número de generaciones que un individuo ha permanecido "vivo". En otras palabras, la edad del cromosoma reemplaza el concepto de selección y, puesto que depende de la aptitud del individuo, ejerce influencia sobre el tamaño de la población en cada etapa del proceso.

Hacia una Taxonomía de Técnicas de Control de Parámetros

Eiben y Michalewicz [1999] propusieron la siguiente taxonomía de técnicas de control de parámetros, con base en los aspectos que se suelen tomar en cuenta:

- **¿Qué se cambia?** (p.ej., representación, función de evaluación, operadores, proceso de selección, porcentaje de mutación, etc.).
- **¿Cómo se realiza el cambio?** (p.ej., heurística determinista, heurística basada en retroalimentación o auto-adaptativa).
- **Alcance/Nivel del Cambio** (p.ej., poblacional, individual, etc.).
- **Evidencia en la que se basa el cambio** (p.ej., monitoreo del desempeño, diversidad de la población, etc.).

Hacia una Taxonomía de Técnicas de Control de Parámetros

El aspecto principal de esta taxonomía es el **¿cómo?**, de acuerdo al cual tenemos tres categorías:

1. **Control Determinista de Parámetros:** Toma lugar cuando el valor de un parámetro se altera usando una regla determinista. Esta regla modifica el valor sin usar ninguna retroalimentación del proceso de búsqueda. Usualmente, se usa una variabilidad dependiente del tiempo (o sea, se efectúa un cambio cada cierto número de generaciones).

Hacia una Taxonomía de Técnicas de Control de Parámetros

2. **Control Adaptativo de Parámetros:** Tomar lugar cuando se usa algún tipo de retroalimentación de la búsqueda para determinar la dirección y/o magnitud del cambio de un parámetro. La asignación del valor del parámetro puede involucrar un sistema de asignación de crédito (o recompensas) y la acción del algoritmo evolutivo puede determinar si el nuevo valor persiste o no, o si éste se propagará a través de la población.

Hacia una Taxonomía de Técnicas de Control de Parámetros

3. **Control Auto-Adaptativo de Parámetros:** En este caso, los parámetros a adaptarse se codifican en el cromosoma y se someten a cruza y mutación. Los mejores valores de esos parámetros codificados nos conducen a mejores individuos, lo que, a su vez, hace más probable que éstos sobrevivan y generen hijos que propaguen dichos valores.

Comentarios Finales sobre Control de Parámetros

Un punto debatible respecto al control de parámetros es: ¿cuánto esfuerzo en este sentido vale la pena llevar a cabo? En otras palabras: ¿qué costos computacionales son aceptables? Algunos investigadores argumentan que el control adaptativo, en general, complica sustancialmente la labor de un algoritmo genético y que las mejoras en la calidad de la solución no son significativas como para justificar el costo extra.

Comentarios Finales sobre Control de Parámetros

Resulta claro que los mecanismos de control adaptativo y auto-adaptativo tienen un costo asociado. Por ejemplo, requieren que se colecten estadísticas durante una corrida o que se realicen operaciones adicionales sobre los individuos. En consecuencia, realizar comparaciones entre un algoritmo que usa auto-adaptación (o adaptación en línea) contra otro que no lo haga, no es algo justo, porque ignora el tiempo que se dedica al ajuste de los parámetros.

Comentarios Finales sobre Control de Parámetros

Los mecanismos de control de parámetros en línea pueden tener gran utilidad en ambientes no estacionarios. En ese caso, frecuentemente es necesario modificar la solución actual debido a varios cambios del ambiente. La capacidad de un algoritmo genético de considerar tales cambios y de rastrear eficientemente el óptimo, ha sido estudiada por varios autores (p.ej., [Angeline et al., 1996], [Vavak et al., 1997]).

Software para Computación Evolutiva

En general, podemos clasificar el software para computación evolutiva en 3 grandes grupos:

- Orientado a las aplicaciones
- Orientado a los algoritmos
- Cajas de herramientas

Software para Computación Evolutiva

- **Orientado a las aplicaciones** : Son "cajas negras" diseñadas para ocultar detalles de los AGs y ayudar al usuario a desarrollar aplicaciones para dominios específicos tales como las finanzas, la programación de horarios, etc.

Software para Computación Evolutiva

- **Orientado a los algoritmos** : Se basan en modelos de AGs específicos y pueden subdividirse en 2 grupos:
  1) Sistemas Específicos: Contienen un solo algoritmo.
  2) Bibliotecas: Contienen una gama de algoritmos y operadores genéticos que pueden estar disponibles sólo de manera pre-compilada.

Software para Computación Evolutiva

- **Cajas de herramientas** (*tool kits*) : Sistemas de programación que proporcionan muchas utilerías, algoritmos y operadores genéticos que pueden usarse para cualquier tipo de aplicación y que normalmente se proporcionan en forma de código fuente (al menos de manera parcial).

Software para Computación Evolutiva

Se pueden subdividir en 2 grupos:

1) *Sistemas Educativos*: Su objetivo es ayudar a los usuarios novatos a practicar los conceptos de computación evolutiva recién aprendidos. Normalmente estos sistemas tienen un número relativamente pequeño de opciones para configurar un cierto algoritmo.

2) *Sistemas de propósito general*: Proporcionan un rico conjunto de herramientas para programar cualquier tipo de AG y aplicarlo a lo que se desee. En algunos casos, incluso permiten que el usuario experto modifique partes del código de acuerdo a sus propias necesidades.

Software de Dominio Público

Nombre: BUGS (*Better to Use Genetic Systems*)

Descripción: Programa interactivo para demostrar el uso de un algoritmo genético. El usuario desempeña el papel de la función de aptitud y trata de evolucionar una cierta forma de vida artificial (curvas).

Software de Dominio Público

El uso de este programa suele facilitar la comprensión de lo que son los AGs y cómo funcionan para aquellos novatos en el área. Además de demostrar los operadores genéticos fundamentales (selección, cruza y mutación), BUGS permite visualizar el "desvío genético" (*genetic drift*) y la convergencia prematura.

Software de Dominio Público

Lenguaje: C bajo X Windows.

Autor: Joshua Smith (jrs@media.mit.edu)

Disponibilidad:
http://www.aic.nrl.navy.mil/pub/galist/src/BUGS.tar.Z

Software de Dominio Público

Nombre: Genesis

Descripción: Implementación de un AG que tiene gran valor histórico por haber sido el primer programa de su tipo liberado en el dominio público.

Software de Dominio Público

Lenguaje: C para Unix.

Autor: John J. Grefenstette (gref@aic.nrl.navy.mil)

Disponibilidad:
http://www.aic.nrl.navy.mil/pub/galist/src/genesis.tar.Z

Software de Dominio Público

Nombre: GENEsYs

Descripción: Implementación de un AG basada en **GENESIS** que incluye extensiones y nuevas funciones para propósitos experimentales. Por ejemplo, cuenta con selección mediante jerarquías lineales, selección de Boltzmann, selección (+), cruza uniforme, recombinación discreta e intermedia, auto-adaptación de porcentajes de mutación, etc.

Software de Dominio Público

También cuenta con una serie de funciones objetivo, incluyendo las funciones de De Jong, funciones continuas de alto grado de complejidad, una instancia del problema del viajero, funciones binarias y una función fractal. Finalmente, tiene también utilerías para monitorear resultados tales como vaciados de mapas de bit de la población, varianza de las variables objeto y de los porcentajes de mutación, etc.

Software de Dominio Público

Lenguaje: C para Unix.

Autor: Thomas Bäck (baeck@ls11.informatik.uni-dortmund.de)

Disponibilidad:
http://www.aic.nrl.navy.mil/pub/galist/src/GENEsYs-1.0.tar.Z

Software de Dominio Público

Nombre: DGenesis

Descripción: Implementación de un AG distribuido desarrollada a partir de GENESIS 5.0. Corre en una red de estaciones de trabajo operando con Unix. Cada subpoblación es manejada por un proceso Unix y la comunicación entre ellas se efectúa usando sockets de Berkeley Unix.

Software de Dominio Público

Lenguaje: C para Unix.

Autor: Erick Cantú Paz (cantupaz@dirichlet.llnl.gov)

Disponibilidad:
http://www.aic.nrl.navy.mil/pub/galist/src/dgenesis-1.0.tar.Z

Software de Dominio Público

Nombre: GECO (*Genetic Evolution through Combination of Objects*)

Descripción: Ambiente de trabajo orientado a objetos para implementar prototipos de algoritmos genéticos. Usa el CLOS (*Common LISP Object System*) y cuenta con abundante documentación y algunos ejemplos de uso.

Software de Dominio Público

Lenguaje: Common LISP para Macintosh o Unix.

Autor: George P. Williams, Jr. (george.p.williams@boeing.com)

Disponibilidad:
http://www.aic.nrl.navy.mil/pub/galist/src/GECO-v2.0.tar.Z

Software de Dominio Público

Nombre: GALOPPS (*Genetic Algorithm Optimized for Portability and Parallelism*)

Descripción: Un sistema de AGs paralelos en el que usuario puede escoger:

- El tipo de problema (con valores numéricos o permutaciones)

Software de Dominio Público

- El tipo de cruza (de entre 7 posibles) y mutación (de entre 4 posibles)
- El tipo de selección (de entre 6 posibles)
- Probabilidades de los operadores, escalamiento de la función de aptitud, frecuencia y patrones de migración

Software de Dominio Público

- Criterios de detención
- Elitismo (opcional)
- Uso de diferente representación para cada subpoblación, con transformación de los migrantes

Software de Dominio Público

- Inversión al nivel de subpoblaciones
- Control sobre el reemplazo poblacional, incluyendo "crowding" y reemplazo aleatorio
- Selección de parejas, usando prevención de incestos

Software de Dominio Público

- Selección de migrantes

  El usuario puede definir una función objetivo (usando una plantilla) y cualquier función auxiliar que necesite. El sistema puede correr una o varias subpoblaciones, en una o varias PCs, estaciones de trabajo o Macs. El sistema corre de manera interactiva (con una interfaz gráfica o de texto) o desde archivos.

Software de Dominio Público

Puede interrumpirse y recomenzarse fácilmente. Existe una versión en PVM que incluso mueve los procesos automáticamente cuando una estación de trabajo está ocupada. Viene con 14 ejemplos que incluyen las funciones de De Jong, la carretera real, el viajero, etc.

Software de Dominio Público

Lenguaje: C para Unix.

Autor: Erik D. Goodman (goodman@egr.msu.edu)

Disponibilidad: http://GARAGE.cps.msu.edu/

Software de Dominio Público

Nombre: ESCaPaDE

Descripción: Sofisticado sistema que permite correr experimentos con algoritmos evolutivos tales como la estrategia evolutiva. El sistema cuenta con 2 tablas internas: una de funciones objetivo y una de monitores de datos, lo que permite una fácil implementación de funciones para monitorear todo tipo de información dentro del algoritmo evolutivo.

Software de Dominio Público

Lenguaje: C para Unix (con rutinas en FORTRAN)

Autor: Frank Hoffmeister
(hoffmeister@ls11.informatik.uni-dortmund.de)

Disponibilidad: Por e-mail a Hoffmeister

Software de Dominio Público

Nombre: GANNET (*Genetic Algorithm/Neural NETwork*)

Descripción: Paquete de software que permite evolucionar redes neuronales binarias. Ofrece toda una variedad de opciones de configuración relacionadas con los valores de los operadores genéticos.

Software de Dominio Público

La evolución de las redes neuronales se basa en 3 funciones de aptitud: precisión entre las entradas y salidas, "estabilidad" de la salida y tamaño de la red. Soporta redes con entradas y salidas binarias, con neuronas de 2 ó 4 entradas y pesos de entre -3 a +4, permitiendo hasta 250 neuronas en una red.

Software de Dominio Público

Lenguaje: C para Unix (con rutinas en FORTRAN)

Autor: Jason Spofford

Disponibilidad: http://fame.gmu.edu/~dduane/thesis

Software de Dominio Público

Nombre: GENOCOP (*Genetic Algorithm for Numerical Optimization for COnstrained Problems*)

Descripción: Paquete de optimización numérica para funciones con cualquier cantidad de restricciones lineales (igualdades y desigualdades).

Software de Dominio Público

Lenguaje: C para Unix.

Autor: Zbigniew Michalewicz (zbyszek@uncc.edu)

Disponibilidad:
http://www.aic.nrl.navy.mil/pub/galist/src/genocop.tar.Z

Software de Dominio Público

Nombre: GPC++

Descripción: Biblioteca de clases en C++ para desarrollar aplicaciones de programación genética. Esta biblioteca define una jerarquía de clases y uno de sus componentes integrales es la capacidad de producir funciones definidas automáticamente.

Software de Dominio Público

Lenguaje: C++ para Unix/MSDOS.

Autor: Thomas Weinbrenner
(thomasw@thor.emk.e-technik.th-darmstadt.de)

Disponibilidad:
http://www.emk.e-technik.thdarmstadt.de/~thomasw/gp.html

Software de Dominio Público

Nombre: GPEIST (*Genetic Programming Environment in SmallTalk*)

Descripción: Ambiente de programación genética en Smalltalk que puede correrse en HP/Sun/PC. Permite distribución de subpoblaciones en varias estaciones de trabajo (con intercambios entre ellas a ciertos intervalos)

Software de Dominio Público

Lenguaje: Smalltalk

Autor: Tony White (arpw@bnr.ca)

Disponibilidad: ENCORE (*The EvolutioNary COmputation REpository network*)

URL: http://www.cs.bham.ac.uk/Mirrors/ftp.de.uu.net/EC/clife/

Software de Dominio Público

Nombre: PGAPack

Descripción: Biblioteca de propósito general para desarrollar AGs paralelos. Incluye:

- Capacidad de invocación desde FORTRAN o C.

Software de Dominio Público

- Soporte de redes de estaciones de trabajo, arquitecturas en paralelo y uniprocesadores.
- Tipos de datos binarios, enteros, reales y caracteres (nativos).
- Soporte de nuevos tipos de datos.

Software de Dominio Público

- Interfaz fácil de usar.
- Niveles múltiples de acceso para usuarios expertos.
- Facilidades extensivas de depuración.

Software de Dominio Público

- Gran cantidad de ejemplos.
- Detallada guía del usuario.
- Soporte de diferentes tipos de selección, cruza y mutación.

Software de Dominio Público

Lenguaje: C para Unix.

Autor: David Levine (levine@mcs.anl.gov)

Disponibilidad: http://www.mcs.anl.gov/pgapack.html

Software de Dominio Público

Nombre: REGAL (*RElational Genetic Algorithm Learner*)

Descripción: Sistema distribuido basado en AGs diseñado para aprender descripciones de conceptos en lógica de primer orden a partir de ejemplos. Se basa en un operador llamado "Sufragio Universal" que permite la probable convergencia asintótica de la población a un estado de equilibrio en el que coexisten varias especies.

Software de Dominio Público

Lenguaje: C para Unix, usando PVM y Tcl/Tk

Autor: Attilio Giordana (attilio@di.unito.it)

Disponibilidad: ftp://ftp.di.unito.it/pub/MLprog/REGAL3.2

Software de Dominio Público

Nombre: SCS-C (*Simple Classifier System in C*)

Descripción: Versión en C del Sistema Clasificador Simple proporcionado en el libro de Goldberg.

Software de Dominio Público

Lenguaje: C para Unix

Autor: Jörg Heitkoetter (joke@de.uu.net)

Disponibilidad: ENCORE

Software Comercial

Nombre: ActiveGA

Descripción: Un OLE que usa un algoritmo genético para solucionar un problema dado. Algunas de sus funciones incluidas son:

- Selección de torneo o ruleta.

Software Comercial

- Parámetros del algoritmo genético definidos por el usuario.
- Invisible durante tiempo de ejecución.
- Ejemplos en Excel, Visual BASIC y Visual C++

Precio: $ 99 dólares

Software Comercial

Nombre: Evolver

Descripción: Paquete de algoritmos genéticos para Windows. Los principiantes pueden usar el módulo para Excel en sus problemas.

Software Comercial

Los usuarios avanzados pueden usar el API incluido para desarrollar sus propias aplicaciones, las cuales pueden ser monitoreadas en tiempo real usando el *EvolverWatcher*.

Precio: $ 349 dólares

Software Comercial

Nombre: PC-Beagle

Descripción: Programa que examina una base de datos con ejemplos y usa técnicas de aprendizaje de máquina para crear un conjunto de reglas de decisión que clasifiquen los ejemplos.

Software Comercial

El sistema contiene 6 componentes principales, de los cuales uno usa algoritmos genéticos.

Precio: £69

Software Comercial

Nombre: MicroGA

Descripción: Herramienta que permite la integración de algoritmos genéticos en cualquier pieza de software. Se trata de un ambiente de programación en C++ que viene en código fuente con documentación y 3 ejemplos completos.

Software Comercial

También incluye un generador de código que permite crear aplicaciones completas de manera interactiva y sin escribir una sola línea de código. Soporte para Macintosh y MS Windows.

Precio: $ 249 dólares

Software Comercial

Nombre: GEATbx (*Genetic and Evolutionary Algorithm Toolbox*)

Descripción: Conjunto de funciones en MATLAB que permiten implementar diferentes tipos de algoritmos genéticos para optimización con uno o varios objetivos.

Software Comercial

- Soporta diferentes tipos de selección (universal estocástica, torneo, jerarquías lineales y no lineales, etc.).
- Incorpora diferentes técnicas de cruza (un punto, dos puntos, uniforme, intermedia, discreta, etc.).
- Incluye mutación para representación entera, real y binaria.

Software Comercial

- Permite diferentes modelos poblacionales (globales, regionales y locales).
- Permite el monitoreo de almacenamiento de resultados (análisis en línea y fuera de línea).
- Cuenta con diversas funciones de prueba.

Software Comercial

- Cuenta con documentación y un tutorial (en HTML).
- Permite la incorporación de conocimiento específico del dominio.

Precio: 150 – 250 euros.

Operadores Avanzados

- Diploides y Dominancia
- Inversión
- Micro-operadores

Diploides y Dominancia

- En AGs, usamos normalmente cromosomas haploides.
- En la naturaleza, sin embargo, los genotipos suelen ser diploides y contienen uno o más pares de cromosomas (a los que se les llama **homólogos**), cada uno de los cuales contiene información (redundante) para las mismas funciones.

Diploides y Dominancia

Ejemplo de un cromosoma diploide:

AbCDefGhIj
aBCdeFgHij

Si suponemos que los genes representados por letras mayúsculas son los dominantes y los representados mediante letras minúsculas son los recesivos, entonces el fenotipo correspondiente al cromosoma anterior sería:

ABCDeFGHIj

Diploides y Dominancia

El operador utilizado en el ejemplo anterior se denomina **dominancia**.

La idea es que un alelo (o un gen) dominante toma precedencia sobre uno recesivo (por ejemplo, los ojos negros son un alelo dominante y los azules uno recesivo).

A un nivel más abstracto, podemos concebir a la dominancia como un mapeo reductor del genotipo hacia el fenotipo.

Diploides y Dominancia

¿Por qué usa esta redundancia la naturaleza?

- Realmente no se sabe.
- Las teorías biológicas más aceptadas, sugieren que los diploides son una especie de "registro histórico" que protegen ciertos alelos (y combinaciones de ellos) del daño que puede causar la selección en un ambiente hostil.

Diploides y Dominancia

En AGs, los diploides suelen usarse para mantener soluciones múltiples (al mismo problema), las cuales pueden conservarse a pesar de que se exprese sólo una de ellas.

La idea es la misma que en Biología: preservar soluciones que fueron efectivas en el pasado, pero que eliminó el mecanismo de selección del AG.

Diploides y Dominancia

Los diploides parecen ser particularmente útiles en problemas en los que el ambiente cambia con el paso de las generaciones (por ejemplo, optimización de funciones dinámicas).

Diploides y Dominancia

El siguiente ejemplo se debe a Hillis (1990, 1992):

Padre 1 (diploide):

A: 10110101 | 01111001111001001010100 1

B: 00000101 | 00100111001111001010100 1

Padre 2 (diploide): ↓

C: 000001110000001111 10 | 000010101011

D: 1111111100001010110 1 | 010111011100

Hijo (diploide): ↓

10110101001001110011111001010100 1

00000111000000111110010111011100

Diploides y Dominancia

El genotipo de un individuo en este ejemplo consiste de 15 pares de cromosomas (por claridad, sólo un par por cada padre se muestra en esta figura). Se elige aleatoriamente un punto de cruza para cada par, y se forma un gameto tomando los alelos antes del punto de cruza en el primer cromosoma, y los alelos después del punto de cruza en el segundo. Los 15 gametos de un padre se unen con los 15 gametos del otro padre para formar un nuevo individuo diploide (nuevamente por claridad sólo un gameto se muestra en esta figura).

Inversión

Holland (1975) propuso formas de adaptar la codificación de su algoritmo genético original, pues advirtió que el uso de cruza de un punto no trabajaría correctamente en algunos casos.

Inversión

El operador de inversión es un operador de reordenamiento inspirado en una operación que existe en genética. A diferencia de los AGs simples, en genética la función de un gene es frecuentemente independiente de su posición en el cromosoma (aunque frecuentemente los genes en un área local trabajan juntos en una red regulatoria), de manera que invertir parte del cromosoma retendrá mucha (o toda) la "semántica" del cromosoma original.

Inversión

Para usar inversión en los AGs, tenemos que encontrar la manera de hacer que la interpretación de un alelo sea la misma sin importar la posición que guarde en una cadena. Holland propuso que a cada alelo se le diera un índice que indicara su posición "real" que se usaría al evaluar su aptitud.

Inversión

Por ejemplo, la cadena 00010101 se codificaría como:

{ (1,0) (2,0) (3,0) (4,1) (5,0) (6,1) (7,0) (8,1) }

en donde el primer elemento de cada uno de estos pares proporciona la posición "real" del alelo dado.

Inversión

La inversión funciona tomando dos puntos (aleatoriamente) a lo largo de la cadena, e invirtiendo la posición de los bits entre ellos. Por ejemplo, si usamos la cadena anterior, podríamos escoger los puntos 3 y 6 para realizar la inversión; el resultado sería:

{ (1,0) (2,0) (6,1) (5,0) (4,1) (3,0) (7,0) (8,1) }

Inversión

Esta nueva cadena tiene la misma aptitud que la anterior porque los índices siguen siendo los mismos. Sin embargo, se han cambiado los enlaces alélicos. La idea de este operador es producir ordenamientos en los cuales los esquemas benéficos puedan sobrevivir con mayor facilidad.

Inversión

Por ejemplo, supongamos que en el ordenamiento original el esquema 00**01** es muy importante. Tras usar este operador, el esquema nuevo será 0010****.

Si este nuevo esquema tiene una aptitud más alta, presumiblemente la cruza de un punto lo preservará y esta permutación tenderá a diseminarse con el paso de las generaciones.

Inversión

Debe advertirse que el uso de este operador introduce nuevos problemas cuando se combina con la cruza de un punto.

Supongamos, por ejemplo, que se cruzan las cadenas:

{ (1,0) (2,0) (6,1) (5,0) (4,1) (3,0) (7,0) (8,1) }

y

{ (5,1) (2,0) (3,1) (4,1) (1,1) (8,1) (6,0) (7,0) }

Inversión

Si el punto de cruza es la tercera posición, los hijos producidos serán:

$$\{ \quad (1,0) \quad (2,0) \quad (6,1) \quad (4,1) \quad (1,1) \quad (8,1) \quad (6,0) \quad (7,0) \quad \}$$

y

$$\{ \quad (5,1) \quad (2,0) \quad (3,1) \quad (5,0) \quad (4,1) \quad (3,0) \quad (7,0) \quad (8,1) \quad \}$$

Inversión

Estas nuevas cadenas tienen algo mal. La primera tiene 2 copias de los bits 1 y 6 y ninguna copia de los bits 3 y 5. La segunda tiene 2 copias de los bits 3 y 5 y ninguna copia de los bits 1 y 6.

¿Cómo podemos asegurarnos de que este problema no se presente?

Inversión

Holland propuso 2 soluciones posibles:

(1) Permitir que se realice la cruza sólo entre cromosomas que tengan los índices en el mismo orden. Esto funciona pero limitaría severamente la cruza.

Inversión

(2) Emplear un enfoque "amo/esclavo":

Escoger un padre como el amo, y reordenar temporalmente al otro padre para que tenga el mismo ordenamiento que su amo. Usando este tipo de ordenamiento se producirán cadenas que no tendrán redundancia ni posiciones faltantes.

Inversión

La inversión se usó en algunos trabajos iniciales con AGs, pero nunca mejoró dramáticamente el desempeño de un AG. Más recientemente se ha usado con un éxito limitado en problemas de "ordenamiento" tales como el del viajero.

Inversión

Sin embargo, no hay todavía un veredicto final en torno a los beneficios que este operador produce y se necesitan más experimentos sistemáticos y estudios teóricos para determinarlos.

Inversión

Adicionalmente, cualquier beneficio que produzca este operador debe sopesarse con el espacio extra (para almacenar los índices de cada bit) y el tiempo extra (para reordenar un padre antes de efectuar la cruza) que se requiere.

Micro-Operadores

En la Naturaleza, muchos organismos tienen genotipos con múltiples cromosomas. Por ejemplo, los seres humanos tenemos 23 pares de cromosomas diploides. Para adoptar una estructura similar en los algoritmos genéticos necesitamos extender la representación a fin de permitir que un genotipo sea una lista de k pares de cadenas (asumiendo que son diploides).

Micro-Operadores

Pero, ¿para qué tomarnos estas molestias? Holland (1975) sugirió que los genotipos con múltiples cromosomas podrían ser útiles para extender el poder de los algoritmos genéticos cuando se usan en combinación con 2 operadores: la segregación y la traslocación.

Micro-Operadores

- **Segregación** : Para entender cómo funciona este operador, imaginemos el proceso de formación de gametos cuando tenemos más de un par cromosómico en el genotipo. La cruza se efectúa igual que como vimos antes, pero cuando formamos un gameto, tenemos que seleccionar aleatoriamente uno de los cromosomas haploides.

Micro-Operadores

A este proceso de selección aleatoria se le conoce como segregación. Este proceso rompe cualquier enlace que pueda existir entre los genes dentro de un cromosoma, y es útil cuando existen genes relativamente independientes en cromosomas diferentes.

Micro-Operadores

- **Traslocación** : Puede verse como un operador de cruza intercromosómico. Para implementar este operador en un algoritmo genético necesitamos asociar los alelos con su "nombre genético" (su posición), de manera que podamos identificar su significado cuando se cambien de posición de un cromosoma a otro mediante la traslocación.

Micro-Operadores

El uso de este operador permite mantener la organización de los cromosomas de manera que la segregación pueda explotar tal organización.

Micro-Operadores

- La segregación y la traslocación no se han usado mucho en la práctica, excepto por algunas aplicaciones de aprendizaje de máquina (e.g., Schaffer, 1984 y Smith, 1980).