Introducción a la Computación Evolutiva

Carlos A. Coello Coello

carlos.coellocoello@ccinvestav.mx
CINVESTAV-IPN
Evolutionary Computation Group (EVOCINV)
Departamento de Computación
Av. IPN No. 2508, Col. San Pedro Zacatenco
México, D.F. 07360, MEXICO

Clase 6

Aditamentos a los Mecanismos de Selección

Selección de Boltzmann

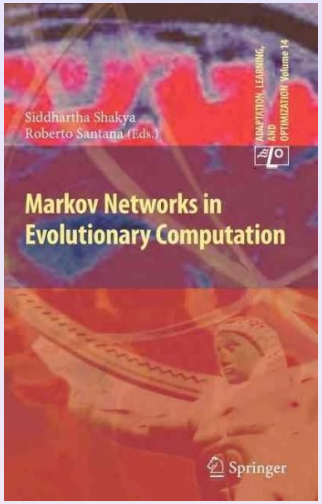
Este esquema de selección se basa en el recocido simulado (*simulated annealing*).

La idea fundamental es usar una función de variación de la "temperatura" para controlar la presión de selección.

Al inicio del proceso evolutivo, se usa un valor alto para la temperatura. Esto hace que la presión de selección sea baja.

Conforme transcurre el proceso evolutivo, la temperatura va disminuyendo, lo cual incrementa la presión de selección.

Aditamentos a los Mecanismos de Selección



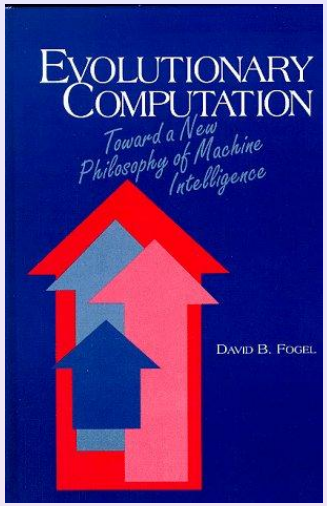
Selección de Boltzmann

Típicamente, se usa la siguiente expresión para calcular el valor esperado de un individuo:

$$Valesp(i,t) = \frac{e^{f(i)/T}}{\langle e^{f(i)/T} \rangle t}$$

donde: T es la temperatura y $\langle\ \rangle^t$ denota el promedio de la población en la generación t.

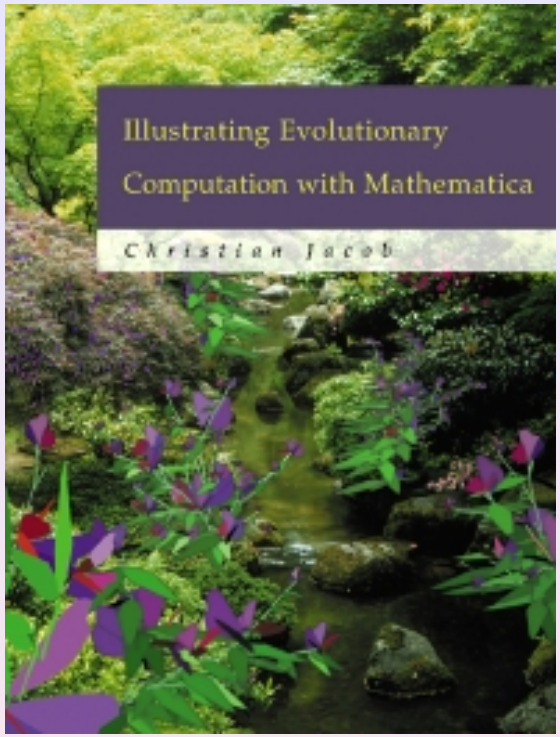
Aditamentos a los Mecanismos de Selección



Selección de Boltzmann

- Se ha utilizado más para optimización multimodal y multiobjetivo (para la formación de nichos).
- Existen pruebas (matemáticas de convergencia de la técnica hacia el óptimo global.
- Tiene el inconveniente de requerir la definición de una función de variación de temperatura.

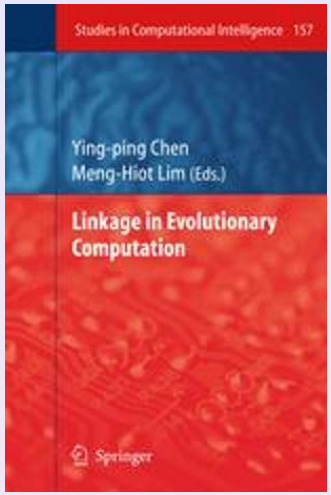
Selección Mediante Torneo



Los métodos de selección proporcional requieren de 2 pasos por generación del algoritmo genético:

1. Calcular la aptitud media (y, la desviación estándar si se usa escalamiento sigma).
2. Calcular el valor esperado de cada individuo.

Selección Mediante Torneo



El uso de jerarquías requiere que se ordene toda la población (una operación cuyo costo puede volverse significativo en poblaciones grandes).

La selección mediante torneo es similar al uso de jerarquías en términos de la presión de selección, pero es computacionalmente más eficiente y más fácil de paralelizarse.

Selección Mediante Torneo

Esta técnica fue propuesta por Wetzel [1983].

La idea básica del método es seleccionar con base en comparaciones directas de los individuos.

Existen 2 versiones de esta técnica de selección:

1. Determinística
2. Probabilística

A. Wetzel, "**Evaluation of the effectiveness of genetic algorithms in combinatorial optimization**", PhD thesis, University of Pittsburgh, Pittsburgh, Philadelphia, USA, 1983.

Selección Mediante Torneo

Algoritmo (versión determinística)

1. Barajar los individuos de la población.
2. Escoger un número P de individuos que participarán en el torneo (típicamente 2).
3. Compararlos con base en su aptitud.
4. El ganador del "torneo" es el individuo más apto.
5. Se debe barajar la población un total de P veces para seleccionar N padres.

Selección Mediante Torneo

Algoritmo (versión probabilística)

La versión probabilística es idéntica a la determinística, excepto por el paso en el que se escoge al ganador.

En este caso, en vez de seleccionar siempre al individuo con aptitud más alta, se aplica *flip*(p) y si el resultado es cierto, se selecciona al más apto. De lo contrario, se selecciona al menos apto.

La probabilidad p permanece fija durante todo el proceso evolutivo y se escoge de manera que:

$$0.5 \leq p \leq 1$$

Selección Mediante Torneo

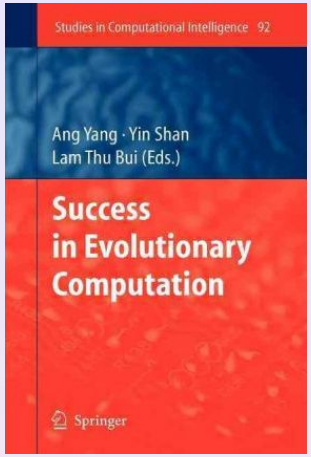
Análisis

La versión determinística garantiza que el mejor individuo será seleccionado P veces.

Cada competencia requiere la selección aleatoria de un número constante de individuos de la población.

La comparación entre estos individuos puede realizarse en tiempo constante y se requieren n competencias de este tipo para completar una generación. Por tanto, el algoritmo es $\mathcal{O}(n)$.

Selección Mediante Torneo



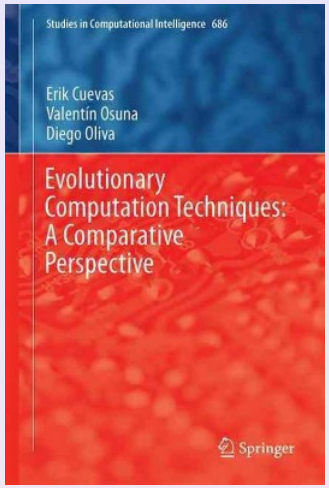
Análisis

Técnica eficiente y fácil de implementar.

No requiere escalamiento de la función de aptitud (usa comparaciones directas).

Puede introducir una presión de selección muy alta porque a los individuos menos aptos no se les da oportunidad de sobrevivir.

Selección Mediante Torneo



Análisis

- Si se usa $t_{torneo} = 1$, se produce una caminata aleatoria con una presión de selección muy baja.
- Si se usa $t_{torneo} = \infty$, la selección se vuelve completamente determinística.
- Si se usa $t_{torneo} \geq 10$, la selección se considera **dura**.
- Si se usa $2 \leq t_{torneo} \leq 5$, la selección se considera **blanda**.

Selección de Estado Uniforme

Fue propuesta por Whitley [1989].

Se usa en los algoritmos genéticos no generacionales, en los cuales sólo uno o unos cuantos individuos son reemplazados en cada generación (los menos aptos).

Esta técnica suele usarse cuando se evolucionan sistemas basados en reglas (p.ej., sistemas de clasificadores) en los que el aprendizaje es incremental.

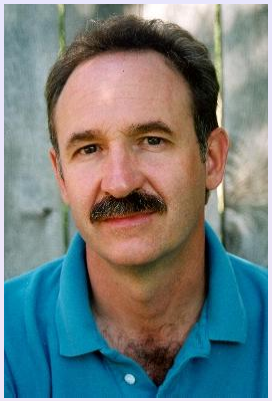
Darrell Whitley, "**The GENITOR Algorithm and Selection Pressure: Why Rank-Based Allocation of Reproductive Trials is Best**", in J. David Schaffer (Editor), *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 116–121, Morgan Kaufmann Publishers, San Mateo, California, USA, 1989.

Selección de Estado Uniforme

Esta técnica es útil cuando los miembros de la población resuelven colectivamente (y no de manera individual) un problema.

Asimismo, los algoritmos genéticos no generacionales se usan cuando es importante “recordar” lo que se ha aprendido antes.

Selección de Estado Uniforme



Algoritmo

1. Seleccionar de G (población original) R individuos ($1 \leq R \leq M$) de entre los más aptos. Normalmente, $R = 1$, o $R = 2$.
2. Efectuar cruza y mutación a los R individuos seleccionados. Llamaremos H a sus hijos.
3. Elegir al mejor individuo en H (o a los S mejores).
4. Reemplazar los S peores individuos de G por los S mejores individuos de H.

Selección de Estado Uniforme

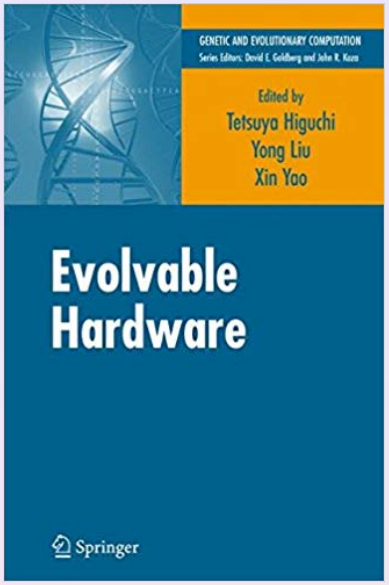
Análisis

Es una técnica especializada de selección.

Su complejidad (en la variante incluida en GENITOR) es $\mathcal{O}(n \log n)$.

Los algoritmos genéticos no generacionales no son muy comunes en aplicaciones de optimización, aunque sí pueden utilizarse en ese dominio.

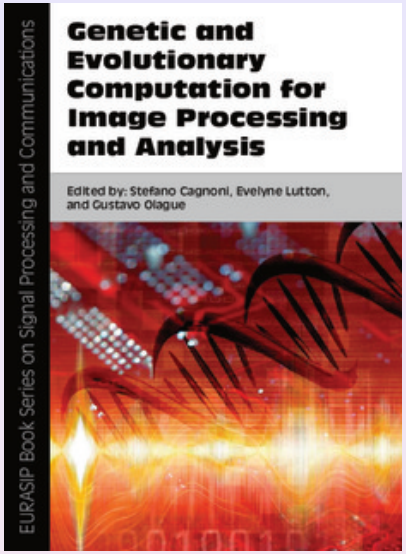
Selección Más (+)



Es también posible en un algoritmo genético usar una selección más (+) como en las estrategias evolutivas. Esta selección consiste en unir la población de padres con la de hijos y seleccionar la mejor mitad de ellos.

Este tipo de selección resulta particularmente útil para resolver problemas de optimización global.

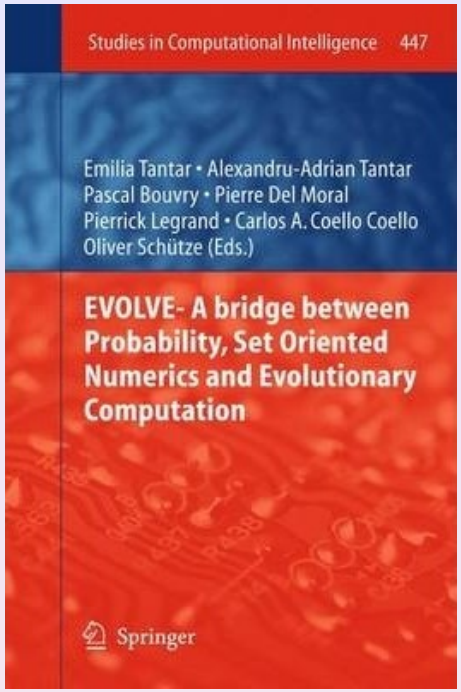
Brecha Generacional



Muy ligada a la selección de estado uniforme se encuentra el concepto de **brecha generacional** *(generation gap)*.

Es importante reconocer en primer término que las poblaciones pueden ser “no traslapables” *(nonoverlapping)* o “traslapables” *(overlapping)*.

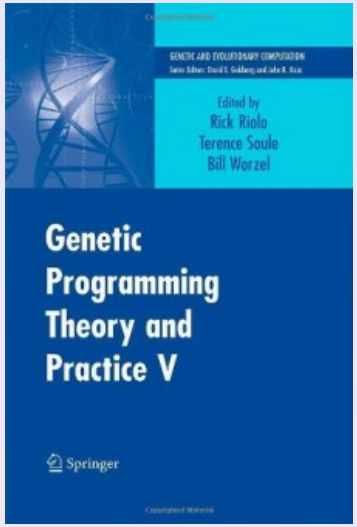
Brecha Generacional



Una población **no traslapable** es aquella en la que los padres nunca compiten contra sus hijos. Es decir, toda la población de padres es siempre reemplazada por la población de hijos.

En una población **traslapable**, los padres compiten contra sus hijos.

Brecha Generacional



Se denomina **brecha generacional** a la cantidad de traslape existente entre padres e hijos. Una brecha generacional grande implica poco (o ningún) traslape poblacional.

Históricamente, la programación evolutiva y las estrategias evolutivas han usado poblaciones *traslapables*, mientras que los algoritmos genéticos han usado poblaciones *no traslapables*.

Brecha Generacional

De Jong [1975] parece haber sido el primero en estudiar algoritmos genéticos con poblaciones traslapables.

De Jong sugirió que las ventajas de las poblaciones traslapables se diluían debido a los efectos negativos del *desvío genético*.

Más tarde, Grefenstette [1986] confirmaría que una brecha generacional mayor parecía mejorar el desempeño del algoritmo genético.

John J. Grefenstette, "**Optimization of Control Parameters for Genetic Algorithms**", *IEEE Transactions on Systems, Man, and Cybernetics*, pp. 122–128, Vol. SMC-16, No. 1, January/February 1986.

Brecha Generacional

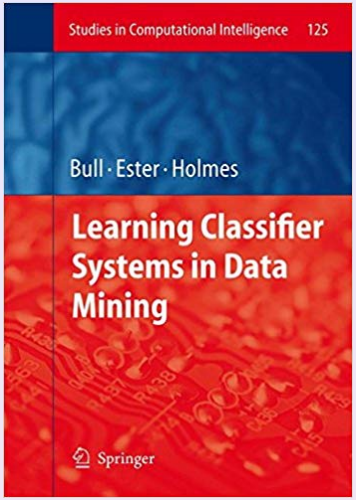
Los primeros experimentos con los *sistemas de clasificadores*, confirmarían, sin embargo, un comportamiento exactamente opuesto [Holland & Reitman, 1978].

En los *sistemas de clasificadores*, el desempeño del algoritmo genético parecía degradarse conforme se aumentaba la brecha generacional.

Algunos investigadores atribuyen los resultados de De Jong y Grefenstette al uso de poblaciones pequeñas.

John H. Holland and Judith H. Reitman, "**Cognitive systems based in adaptive algorithms**", in D.A. Waterman and F. Hayes-Roth (Editors), *Patter-Directed Inference Systems*, pp. 313–329, Academic Press, New York, USA, 1978.

Brecha Generacional

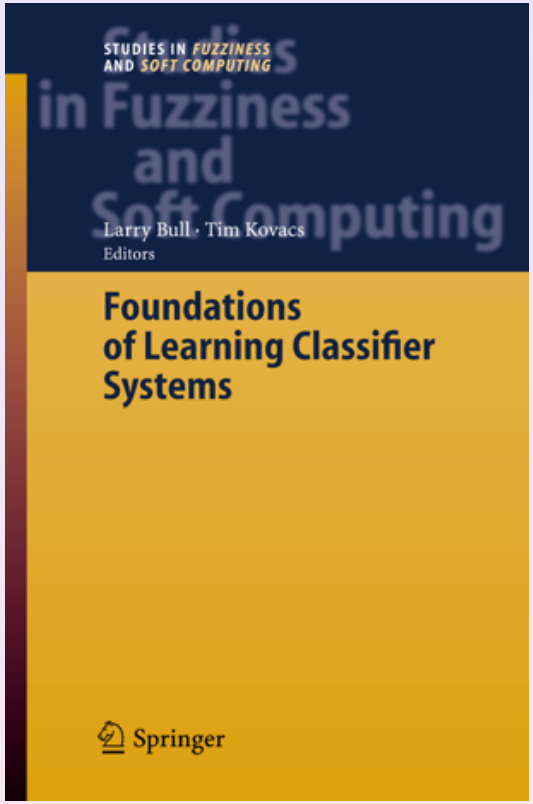


Los algoritmos genéticos tradicionales siguen usando, sin embargo, poblaciones no traslapables.

Los algoritmos genéticos de estado uniforme son aquellos en los que la población es traslapable.

Normalmente, sólo uno o dos hijos se producen en cada iteración de un algoritmo genético de estado uniforme.

Otras Técnicas de Selección



- Disruptiva
- Jerarquías no lineales
- Competitiva

Selección Disruptiva

Propuesta por Kuo y Hwang [1993] para normalizar aptitudes con respecto a un cierto valor moderado (en vez de usar valores extremos).

Suele usarse: $f'_i(x) = |f_i(x) - f(\bar{t})|$ $Valesp_i = \frac{f'_i(x)}{f'(\bar{t})}$, donde $(f(\bar{t}))$ se refiere a la aptitud media de la población.

Ting Kuo and Shu-Yuen Hwang, "**A Genetic Algorithm with Disruptive Selection**", in Stephanie Forrest (Editor), *Proceedings of the Fifth International Conference on Genetic Algorithms*, pp. 65–69, Morgan Kaufmann Publishers, San Mateo, California, USA, July 1993.

Selección Disruptiva

Motivación: distribuir más los esfuerzos de la búsqueda hacia las soluciones extremadamente buenas y extremadamente malas.

Los individuos cercanos a la media son desechados.

La utilidad de este método es altamente dependiente en la aplicación.

Suele usarse con funciones de aptitud dinámicas (aquellas donde la posición del óptimo cambia en el tiempo).

Jerarquías no Lineales

Fue propuesta por Michalewicz [1996]. Se usa:

$$prob_i = q(1-q)^{\text{jerarquía}_i - 1}$$

donde:
$prob_i$ es la probabilidad de que el individuo i sea seleccionado.
$q \in [0 \ldots 1]$ es el factor de presión de selección, jerarquía_i es la jerarquía del individuo i.

Zbigniew Michalewicz, "**Genetic Algorithms + Data Structures = Evolution Programs**", Springer-Verlag, Third Edition, New York, USA, 1996.

Jerarquías no Lineales

Al igual que con las jerarquías lineales, se asigna la jerarquía más baja al peor individuo y la más alta al mejor.

Una vez que se conoce la probabilidad de que un individuo sea seleccionado, podemos calcular su valor esperado multiplicando dicho valor por n (tamaño de la población).

Posteriormente, podemos aplicar cualquier técnica de selección proporcional.

Jerarquías no Lineales

Thomas Bäck advirtió que las probabilidades obtenidas con este método no suman uno.

También notó que la técnica puede hacerse prácticamente idéntica al torneo, dependiendo del valor de q que se use.

Valores grandes de q implican una mayor presión de selección.

Jerarquías no Lineales

Michalewicz [1996] advierte que la suma de probabilidades puede hacerse igual a 1 si usamos:

$$prob_i = c * q(1-q)^{\text{jerarquía}_i - 1}$$

donde:

$$c = \frac{1}{1-(1-q)^M}$$

y M es el tamaño de la población.

Selección Competitiva

En este caso, la aptitud de un individuo se determina mediante sus interacciones con otros miembros de la población, o con otros miembros de una población separada que evoluciona concurrentemente.

Esta técnica ha sido utilizada por Hillis [1992], Angeline & Pollack [1993] y Sebald & Schlenzig [1994].

Puede verse como un esquema co-evolutivo: las aptitudes de dos individuos dependen mutuamente entre sí.

W.D. Hillis, "**Co-evolving parasites improves simulated evolution as an optimization procedure**", in C. Langton, C. Taylor, J. Farmer and S. Rasmussen (Editors), *Artificial Life II*, pp. 313–324, Addison-Wesley, Reading, Massachusetts, USA, 1992.

Peter J. Angeline and Jordan B. Pollack, "**Competitive Environments Evolve Better Solutions for Complex Tasks**", in Stephanie Forrest (Editor), *Proceedings of the Fifth International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers, San Mateo, California, USA, July 1993.

Anthony V. Sebald and Jennifer Schlenzig, "**Minimax design of neural net controllers for highly uncertain plants**", *IEEE Transactions on Neural Networks*, Vol. 5, No. 1, pp. 73–82, January 1994.

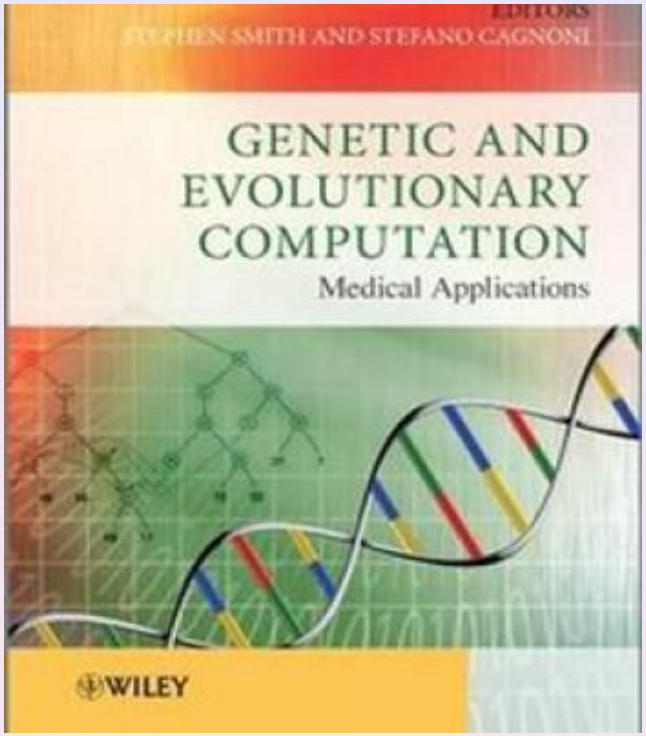
Clasificaciones de Técnicas de Selección

Bäck y Hoffmeister [1991] distinguen entre:

1. **Métodos Estáticos**: Requieren que las probabilidades de selección permanezcan constantes entre generaciones. Ejemplo: jerarquías lineales.
2. **Métodos Dinámicos**: No se requiere que las probabilidades de selección permanezcan constantes. Ejemplo: selección proporcional

Thomas Bäck and Frank Hoffmeister, "**Extended Selection Mechanisms in Genetic Algorithms**", in Richard K. Belew and Lashon B. Booker (Editors), *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 92–99, Morgan Kaufmann Publishers, San Mateo, California, USA, July 1991.

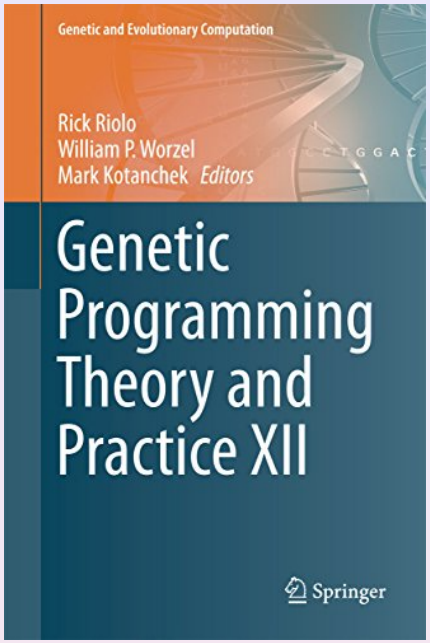
Clasificaciones de Técnicas de Selección



Otros investigadores distinguen entre:

1. **Selección Preservativa**: Requiere una probabilidad de selección distinta de cero para cada individuo.
2. **Selección Extintiva**: Puede asignar una probabilidad de selección de cero a algún individuo.

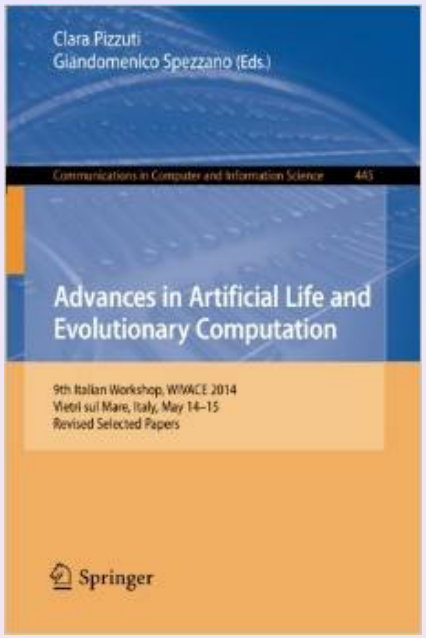
Clasificaciones de Técnicas de Selección



A su vez, las **técnicas extintivas** se dividen en:

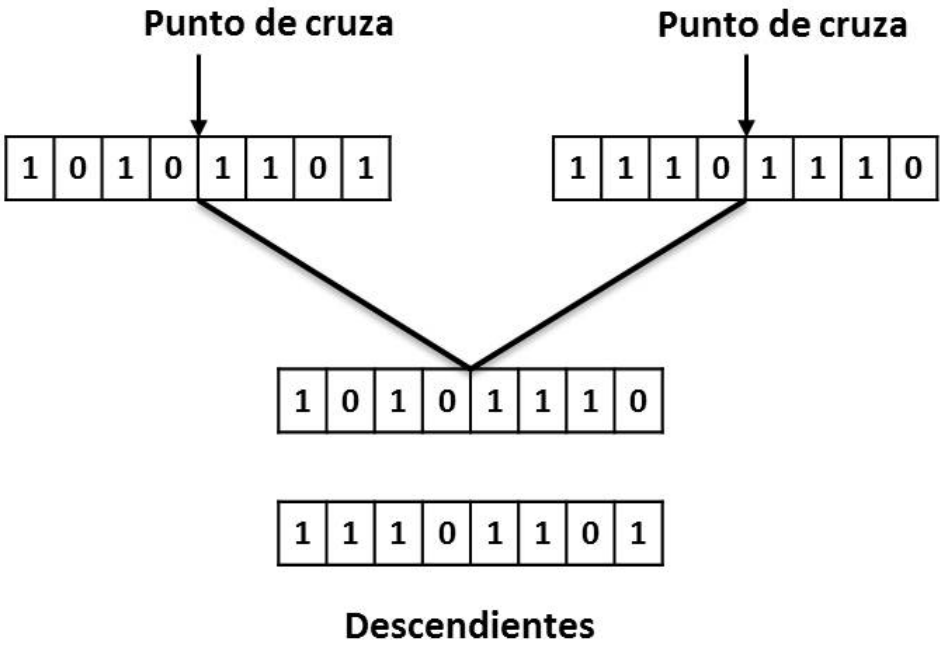
1. **Selección Izquierda**: Se impide a los mejores individuos reproducirse a fin de evitar convergencia prematura.
2. **Selección Derecha**: No se tiene control explícito sobre la capacidad reproductiva de los individuos más aptos.

Clasificaciones de Técnicas de Selección



Adicionalmente, algunas técnicas de selección son **puras** en el sentido de que a los padres se les permite reproducirse solamente en una generación (es decir, el tiempo de vida de cada individuo está limitado a sólo una generación, independientemente de su aptitud).

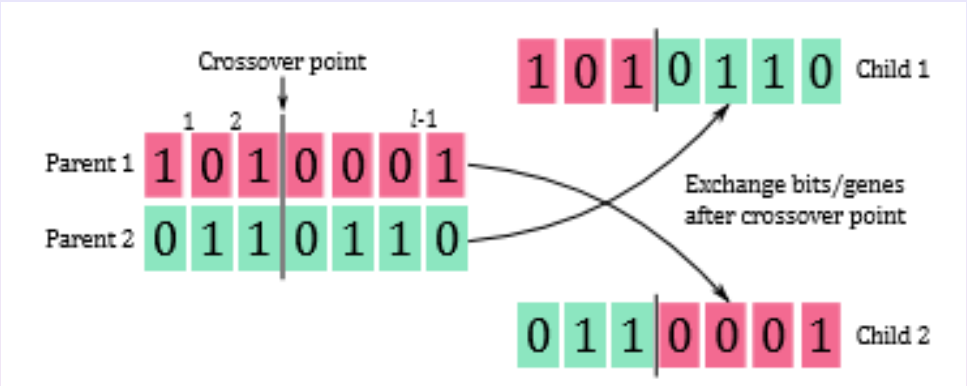
Técnicas de Cruza



En los sistemas biológicos, la cruza es un proceso complejo que ocurre entre parejas de cromosomas.

Estos cromosomas se alinean, luego se fraccionan en ciertas partes y, posteriormente, intercambian fragmentos entre sí.

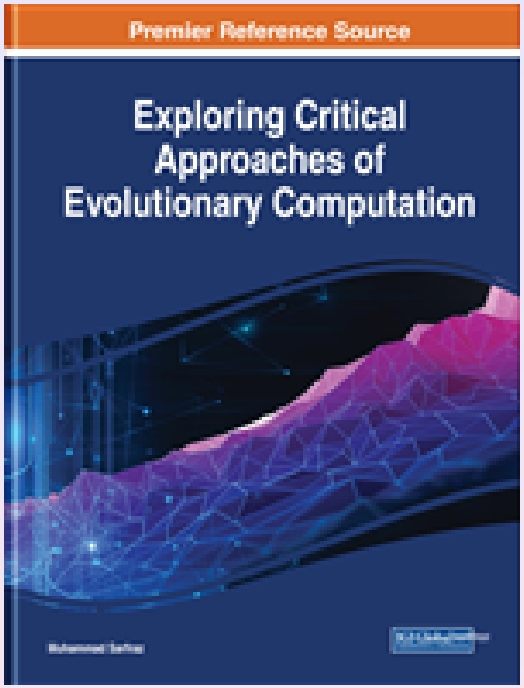
Técnicas de Cruza



En computación evolutiva se simula la cruza intercambiando segmentos de cadenas lineales de longitud fija (los cromosomas).

Aunque hemos visto técnicas de cruza básicas para representación binaria, éstas son generalizables a alfabetos de cardinalidad mayor, si bien en algunos casos requieren de ciertas modificaciones.

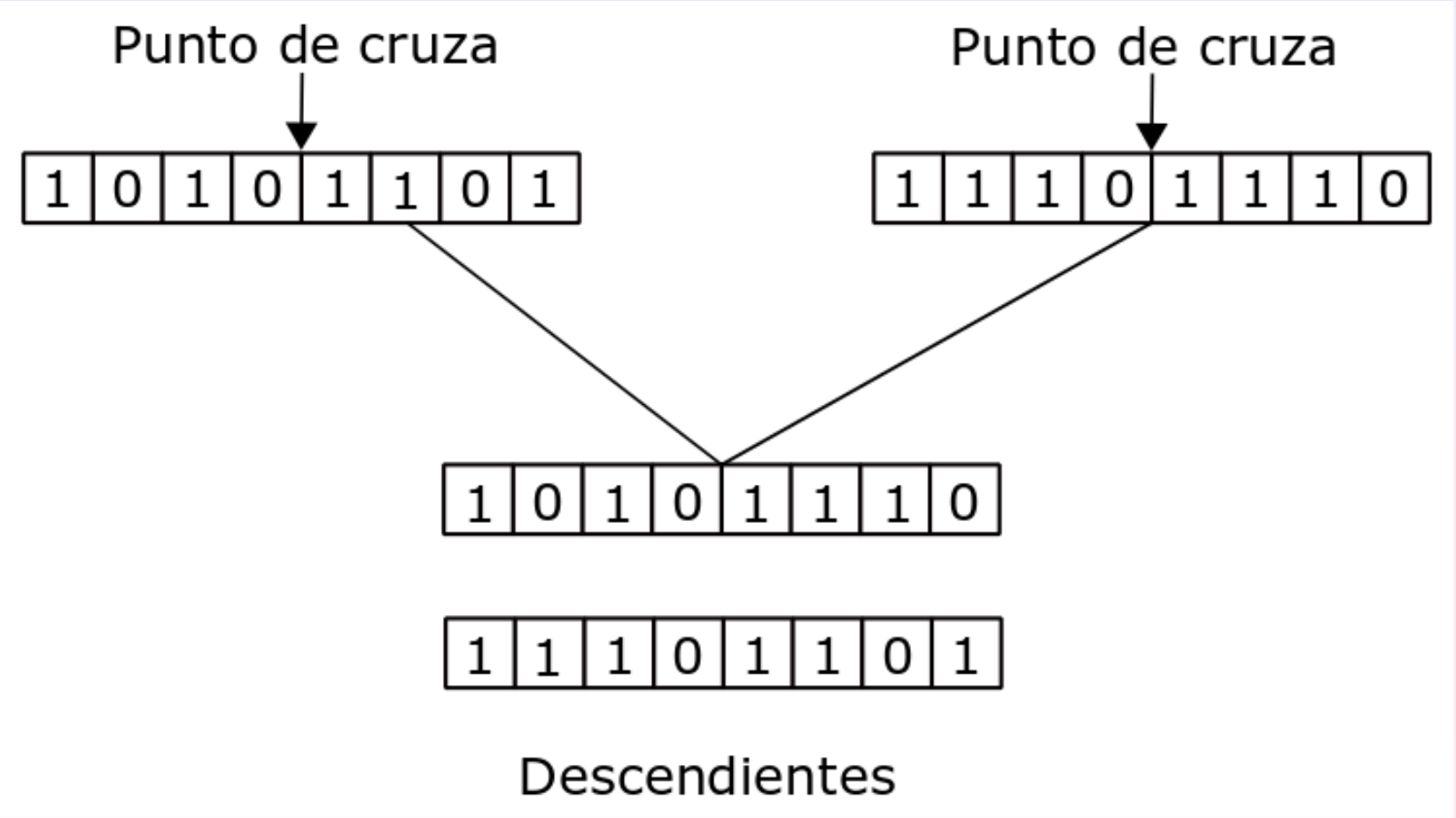
Técnicas de Cruza



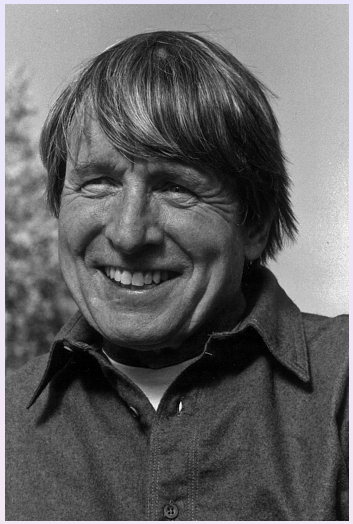
Comenzaremos por revisar las 3 técnicas básicas de cruza:

1. Cruza de un punto
2. Cruza de dos puntos
3. Cruza uniforme

Cruza de un Punto



Cruza de un Punto



Propuesta por Holland [1975].

No suele usarse mucho en la práctica debido a sus limitantes teóricas.

Puede demostrarse, por ejemplo, que hay varios esquemas que no pueden formarse bajo esta técnica de cruza.

Cruza de un Punto

Definamos a δ como la **longitud de definición** de un esquema.

$\delta(H)$ = distancia entre la primera y la última posición fija de un esquema H.

Ejemplo:
$\delta(*11*0*0*) = 7 - 2 = 5$
$\delta(**1*****) = 0$

Cruza de un Punto

La cruza de un punto destruye esquemas en los que la longitud de definición es alta.

Esto produce el denominado “sesgo posicional”: los esquemas que pueden crearse o destruirse por la cruza dependen fuertemente de la localización de los bits en el cromosoma [Eshelman et al., 1989].

Larry J. Eshelman, Richard A. Caruana and J. David Schaffer, “**Biases in the Crossover Landscape**’, in J. David Schaffer (Editor), *Proceedings of the Third International Conference on Genetic Algorithms*, pp. 10–19, Morgan Kaufmann Publishers, San Mateo, California, USA, 1989.

Cruza de un Punto

El problema fundamental de la cruza de un punto es que presupone que los bloques constructores son esquemas cortos y de bajo orden, y cuando esto no sucede (p.ej., con cadenas largas), este operador suele no proporcionar resultados apropiados.

Cruza de un Punto

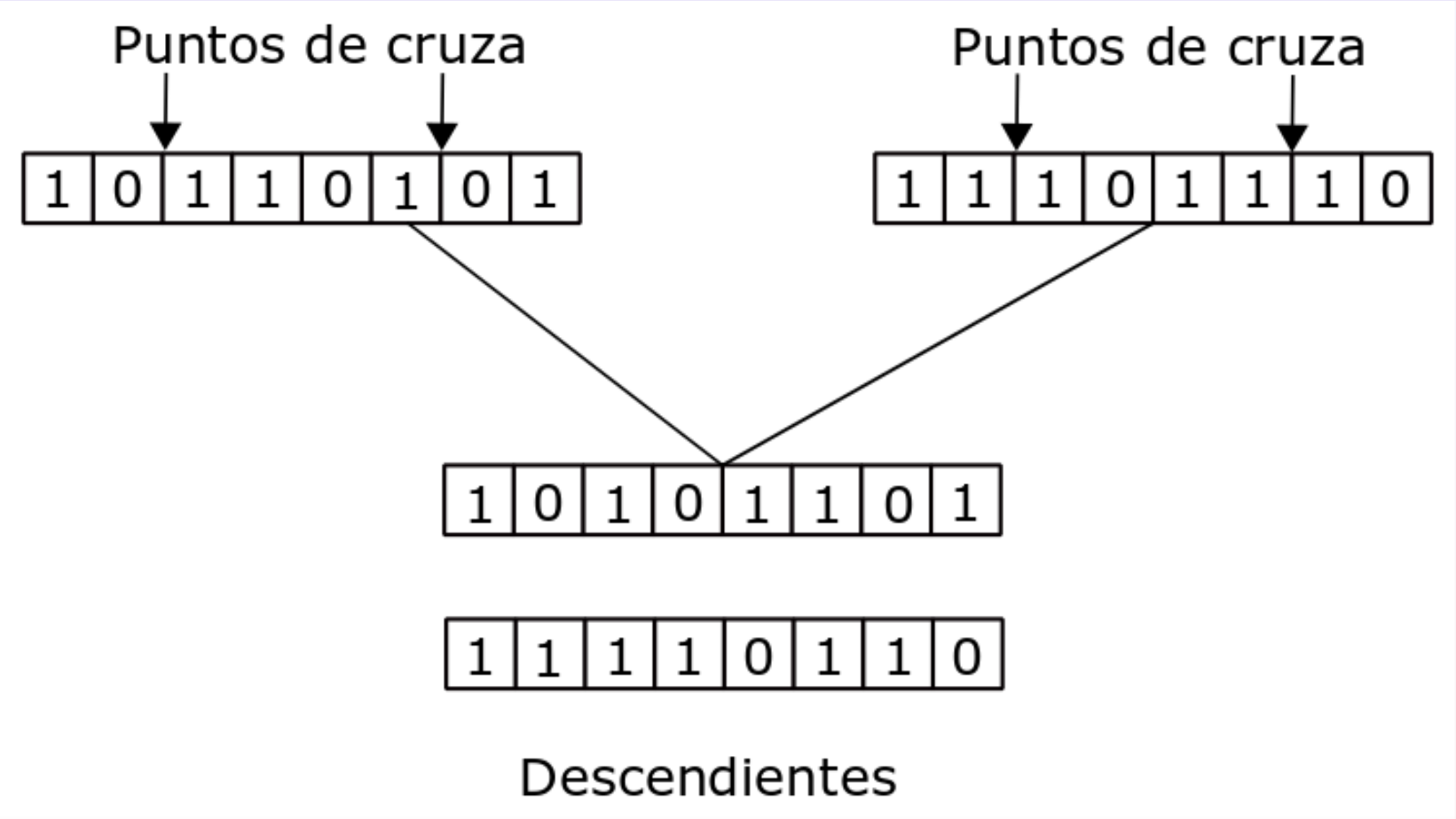
Obviamente, las aplicaciones del mundo real suelen requerir cadenas largas.

La cruza de un punto trata también preferencialmente algunas posiciones del cromosoma, como por ejemplo los extremos de una cadena.

Cruza de un Punto

La cruza de un punto suele preservar también los “hitchhikers”, que son bits que no son parte del esquema deseado, pero que debido a su similitud con ellos gozan de los beneficios de la cruza.

Cruza de Dos Puntos



Cruza de Dos Puntos

De Jong [1975] fue el primero en implementar una cruza de n puntos, como una generalización de la cruza de un punto.

El valor $n = 2$ es el que minimiza los efectos disruptivos (o destructivos) de la cruza y de ahí que sea usado con gran frecuencia.

A. K. De Jong, "**An Analysis of the Behavior of a Class of Genetic Adaptive Systems**", PhD thesis, University of Michigan, Ann Arbor, USA, 1975.

Cruza de Dos Puntos

No existe consenso en torno al uso de valores para n que sean mayores o iguales a 3.

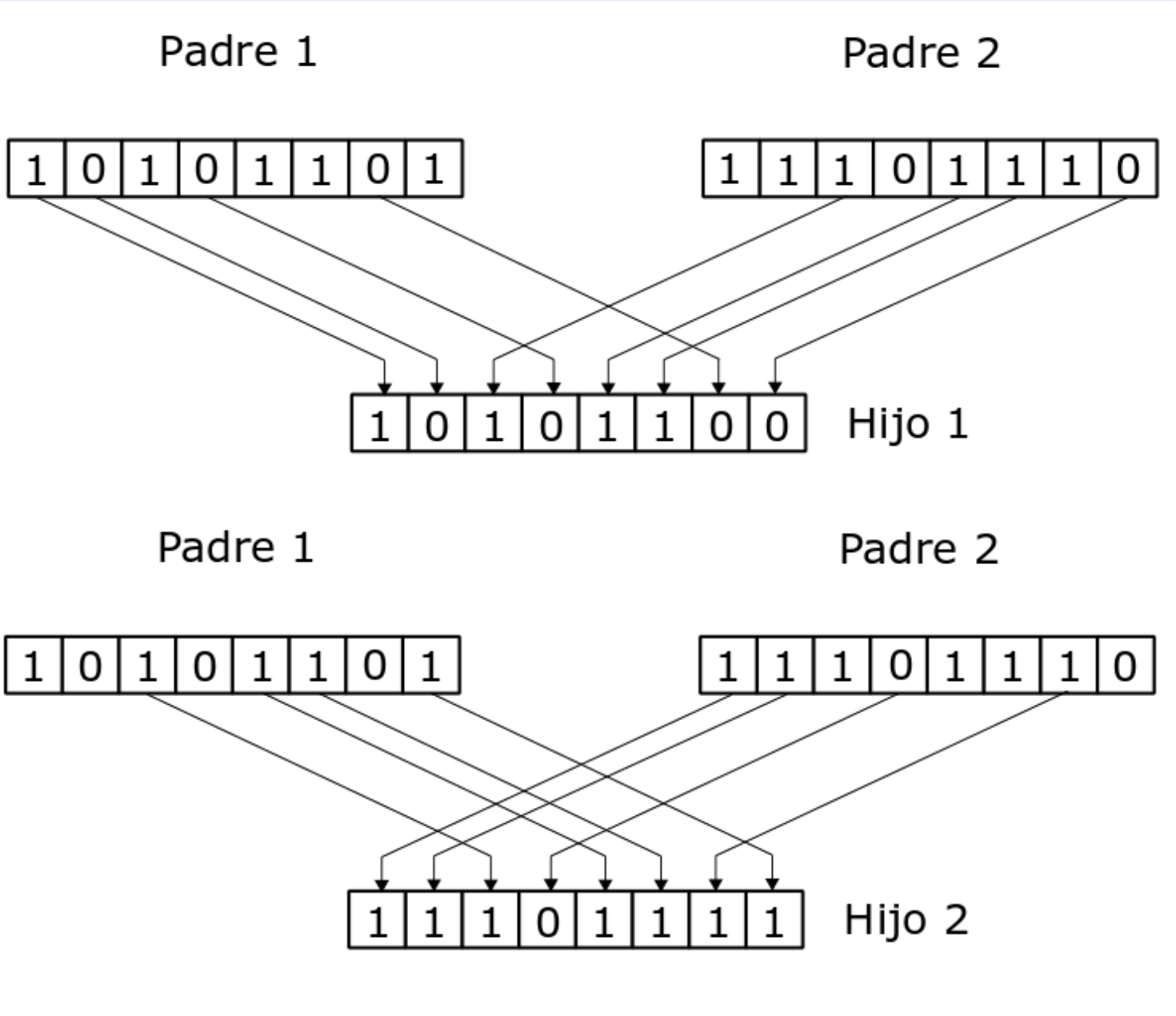
Los estudios empíricos al respecto [De Jong, 1975; Eshelman et al., 1989] proporcionan resultados que no resultan concluyentes respecto a las ventajas o desventajas de usar dichos valores.

Cruza de Dos Puntos

En general, sin embargo, es aceptado que la cruza de dos puntos es mejor que la cruza de un punto.

Asimismo, el incrementar el valor de n se asocia con un mayor efecto disruptivo de la cruza.

Cruza Uniforme



Cruza Uniforme

Fue propuesta originalmente por Ackley [1987], aunque se le suele atribuir a Syswerda [1989].

En este caso, el número de puntos de cruza no se fija previamente.

La cruza uniforme tiene un mayor efecto disruptivo que cualquiera de las dos cruzas anteriores.

David H. Ackley, "**A Connectionist Machine for Genetic Hillclimbing**", Kluwer Academic Publishers, Boston, Massachusetts, USA, 1987.

Gilbert Syswerda, "**Uniform Crossover in Genetic Algorithms**", in J. David Schaffer (Editor), *Proceedings of the Third International Conference on Genetic Algorithms*, pp 2–9, Morgan Kaufmann Publishers, San Mateo, California, USA, 1989.

Cruza Uniforme

Suele usarse con $Pc = 0.5$.

Algunos investigadores, sin embargo, sugieren usar valores más pequeños de Pc [Spears & De Jong, 1991].

Cuando se usa $Pc = 0.5$, hay una alta probabilidad de que todo tipo de cadena binaria de longitud L sea generada como máscara de copiado de bits.

William M. Spears and Kenneth A. De Jong, "**An Analysis of Multi-Point Crossover**", in Gregory E. Rawlins (Editor), *Foundations of Genetic Algorithms*, pp. 301–315, Morgan Kaufmann Publishers, San Mateo, California, USA, 1991.

Cruza Acentuada

Propuesta por Schaffer y Morishima [1987].

En vez de calcular directamente la máscara (o patrón) de cruza, la idea es usar una cadena binaria de "marcas" para indicar la localización de los puntos de cruza.

La idea fue sugerida por Holland [1975], aunque en un sentido distinto.

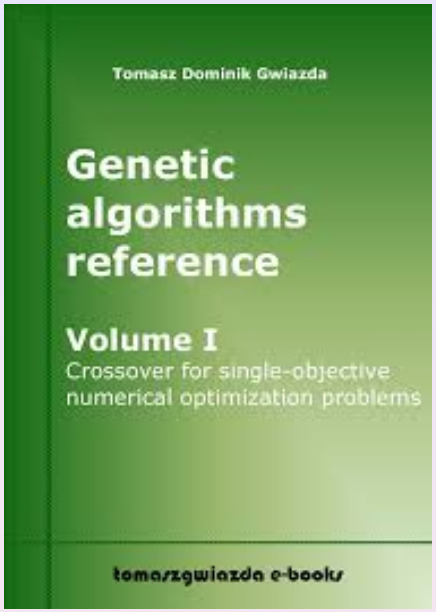
J. David Schaffer and Amy Morishima, "**An Adaptive Crossover Distribution Mechanism for Genetic Algorithms**", in John J. Grefenstette (Editor), *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, pp. 36–48, Lawrence Erlbaum Associates, Hillsdale, New Jersey, USA, July 1987.

Cruza Acentuada

La información extra se agrega al cromosoma de manera que el número y localizaciones de los puntos de cruza pueda ser objeto de manipulación por el algoritmo genético.

Por tanto, las cadenas tendrán una longitud del doble de su tamaño original.

Cruza Acentuada



Marcamos con “1” las posiciones donde hay cruza y con “0” las posiciones donde no la hay.

Se suelen usar signos de admiración (!) para facilitar la escritura de las cadenas.

Cruza Acentuada

Ejemplo:
Cromosoma:

0 1 1 0 0 0 1 1 0 0 : 0 1 0 0 1 0 0 0 0 0

cadena original L = 10 puntos de cruza L = 10

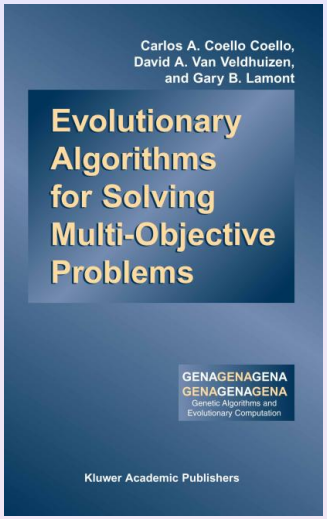
Puede interpretarse como:

01!100!01100

↑ ↑

Aquí se efectúa la cruza

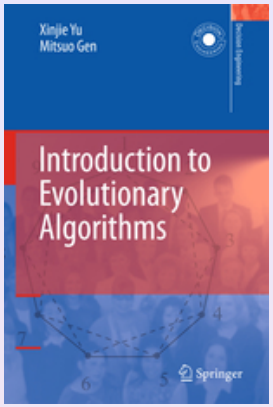
Cruza Acentuada



Algoritmo

1. Copiar los bits de cada padre hacia sus hijos, de uno en uno.
2. En el momento en que se encuentra un signo de admiración en cualquiera de los padres, se efectúa la cruza (es decir, se invierte la procedencia de los bits en los hijos).
3. Cuando esto ocurre, los signos de admiración se copian también a los hijos, justo antes de que la cruza se efectúe.

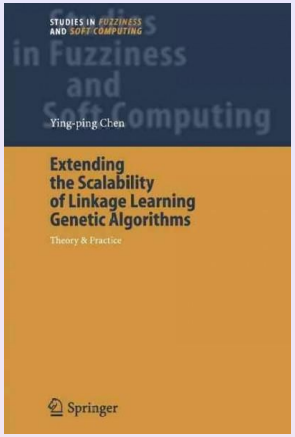
Cruza Acentuada



Ejemplo:
Antes de la cruza:
P1 = a a a a a a a! b b b b b b b
P2 = c c c c! d d d d d d! e e e e

Después de la cruza:
H1 = a a a a d d d b b b e e e e
H2 = c c c c! a a a! d d d! b b b b

Cruza Acentuada

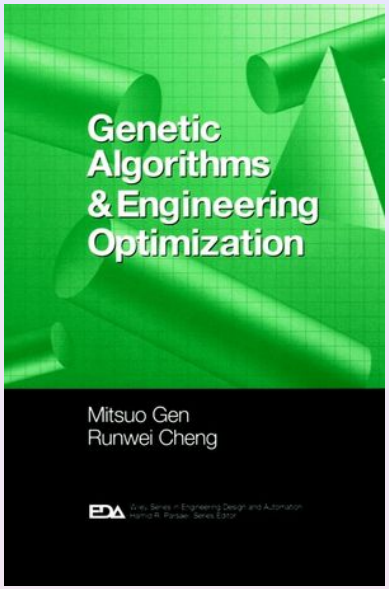


Sólo se usa la primera parte de la cadena para calcular la aptitud, pero se espera que la selección, cruza y mutación tengan un efecto positivo sobre los puntos de cruza.

La mutación actúa sobre los dos segmentos cromosómicos.

Las probabilidades de que aparezcan unos en el segundo segmento se determinan de manera distinta a las del primero.

Cruza Acentuada



La técnica reportó buenos resultados en un pequeño conjunto de funciones de prueba.

Sin embargo, no hay evidencia contundente acerca de su efectividad.

Esta técnica tiene una buena inspiración biológica, porque estas marcas de cruza efectivamente existen en la naturaleza y se co-evolucionan junto con los cromosomas.

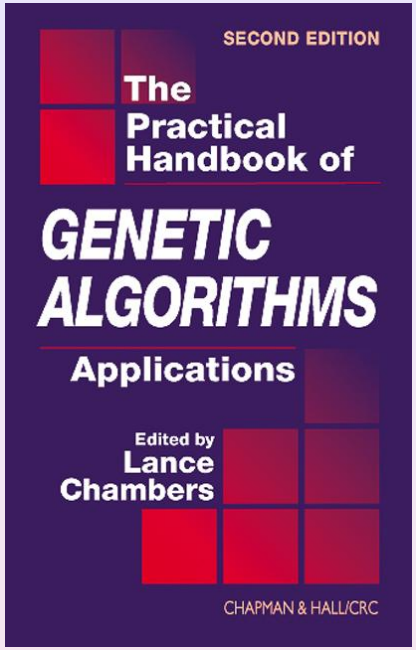
Sesgos de la Cruza

El "**sesgo**" de la cruza se refiere a las tendencias de este operador hacia favorecer o no un cierto tipo de búsqueda.

La búsqueda aleatoria es la única que no presenta ningún tipo de sesgo.

Desde hace algún tiempo, se ha determinado que se requiere de algún tipo de sesgo para que una técnica de búsqueda sea efectiva [Mitchell, 1980].

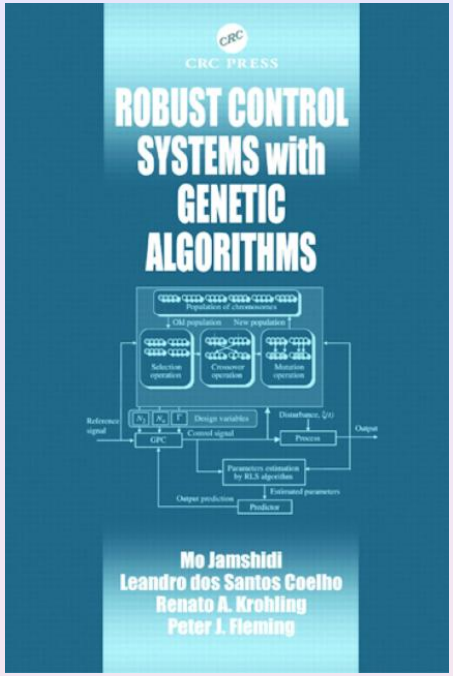
Sesgos de la Cruza



En algoritmos genéticos, se suelen considerar 2 tipos de sesgo para la cruza:

1. Distribucional
2. Posicional

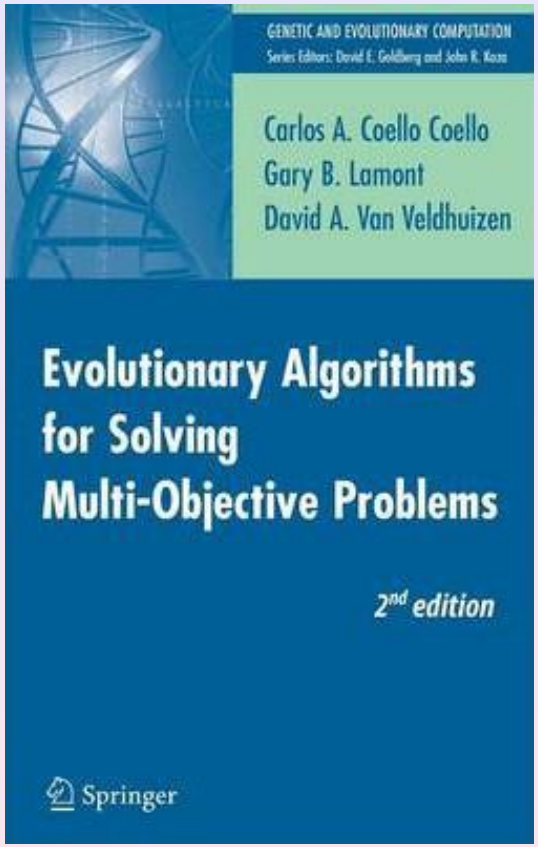
Sesgos de la Cruza



Sesgo Distribucional

El **sesgo distribucional** se refiere al número de símbolos transmitidos durante una recombinación. Asimismo, se refiere a la medida en la que algunas cantidades tienen más tendencia a ocurrir que otras.

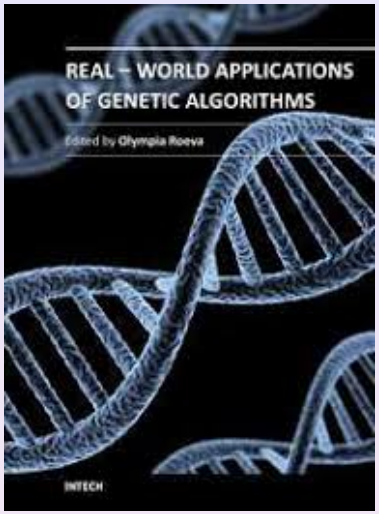
Sesgos de la Cruza



Sesgo Distribucional

El **sesgo distribucional** es importante porque está correlacionado con el número potencial de esquemas de cada padre que pueden ser recombinados por el operador de cruza.

Sesgos de la Cruza



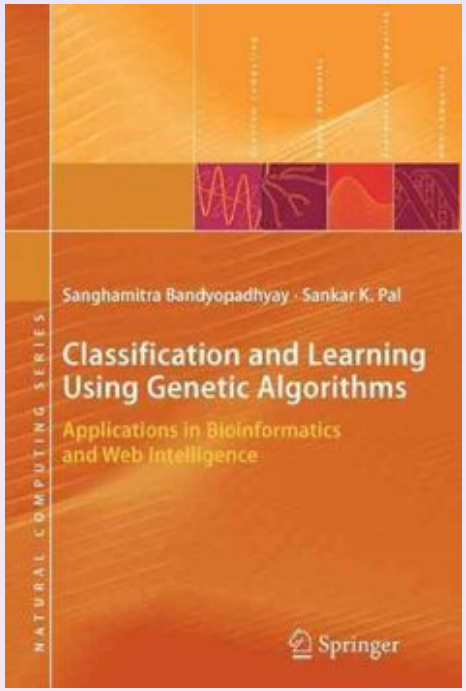
Sesgo Distribucional

La cruza de un punto y la de dos puntos no tienen sesgo distribucional.

La cruza de n puntos ($n > 2$) tiene un sesgo distribucional moderado.

La cruza uniforme tiene un sesgo distribucional muy fuerte.

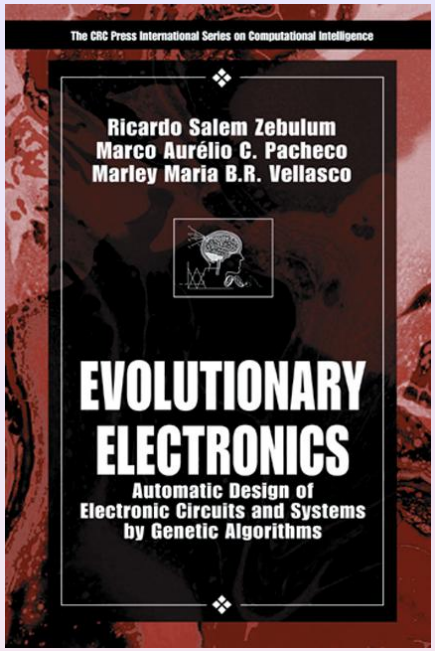
Sesgos de la Cruza



Sesgo Posicional

El **sesgo posicional** caracteriza en qué medida la probabilidad de que un conjunto de símbolos se transmitan intactos durante la recombinación depende de las posiciones relativas de los mismos en el cromosoma.

Sesgos de la Cruza

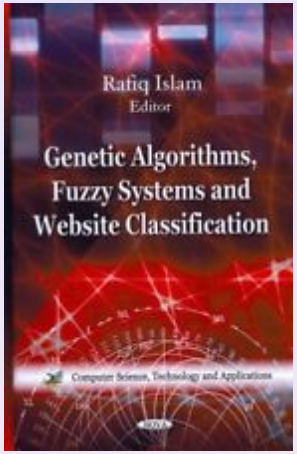


Sesgo Posicional

El **sesgo posicional** es importante porque indica qué esquemas es más probable que se hereden de padres a hijos.

También indica la medida en la que estos esquemas aparecerán en nuevos contextos.

Sesgos de la Cruza



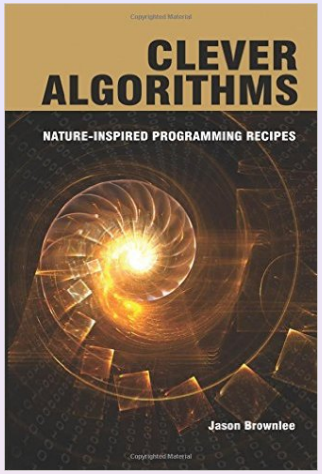
Sesgo Posicional

La cruza de un punto tiene un fuerte sesgo posicional.

Todo parece indicar que la cruza de n puntos tiene también un sesgo posicional fuerte, aunque éste varía en función de n.

La cruza uniforme no tiene sesgo posicional.

Variantes de la Cruza

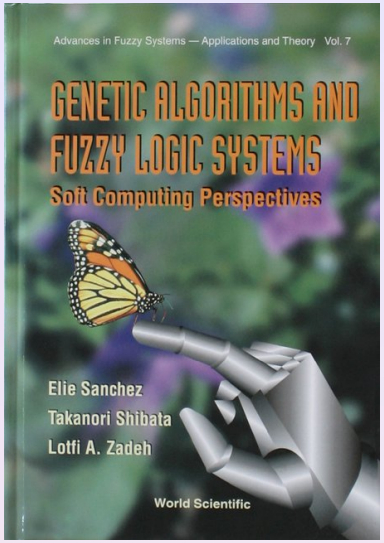


En la práctica, diversos aspectos de la cruza suelen modificarse para mejorar su desempeño.

Una variante, por ejemplo, consiste en retener sólo a uno de los dos hijos producidos por una cruza sexual.

Holland [1975] describe una técnica de este tipo.

Variantes de la Cruza



Estudios empíricos han mostrado, sin embargo, que retener a los 2 hijos producidos por una cruza sexual reduce sustancialmente la pérdida de diversidad en la población [Booker, 1982].

Lashon B. Booker, "**Intelligent Behavior as an Adaptation to the Task Environment**", PhD thesis, Logic of Computers Group, University of Michigan, Ann Arbor, Michigan, USA, 1982.