
Inteligencia Artificial

Algoritmos Genéticos

Jorge Luis Guevara Diaz

www.jorge.sistemasyservidores.com

Algoritmos Genéticos

- Considere los siguientes problemas:
 - Pedro desea visitar 12 universidades localizadas en 10 distintos departamentos, antes de decidir las 5 a las que va a postular, sus padres le dicen a Jose, que dibuje la ruta más eficiente en el mapa para poder así viajar en el auto de la familia
 - Javier desea programar un software que maneje un telescopio, de tal manera que el número de galaxias que pueda observar sea maximizado, mientras el movimiento del telescopio sea minimizado
-

Algoritmos Genéticos

- Ambos problemas consisten en variaciones del TSP (traveling salesperson problem), que es un problema de partida clásico en los algoritmos genéticos
 - La solución óptima será el fitness individual de cada generación, luego de muchos ciclos de selección, reproducción y mutación
-

Algoritmos Genéticos

■ Fundamento biológico

□ Cromosoma

- Los organismos vivos tienen **células**, en cada uno existe un conjunto de **cromosomas**
 - Son cadenas de **ADN**, y son como un modelo para el organismo
 - Los **cromosomas** son un conjunto de **genes** (bloques de **ADN**), cada gen codifica **proteínas** particulares, (rasgos como el color de los ojos)
 - Las posibles ajustes para un **gen** se llaman **alelos** (color de ojos = marron)
-

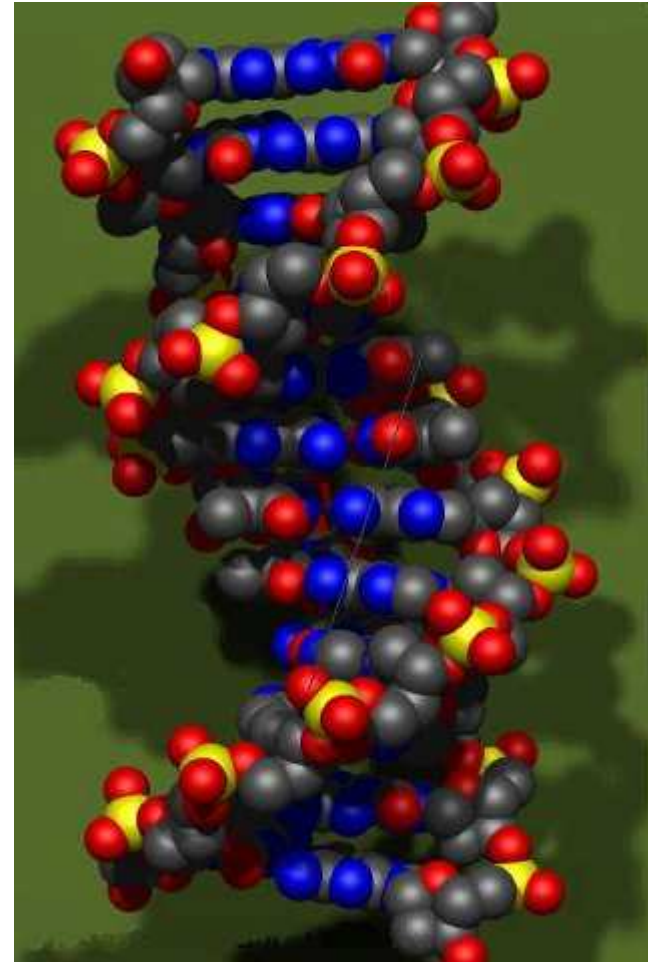
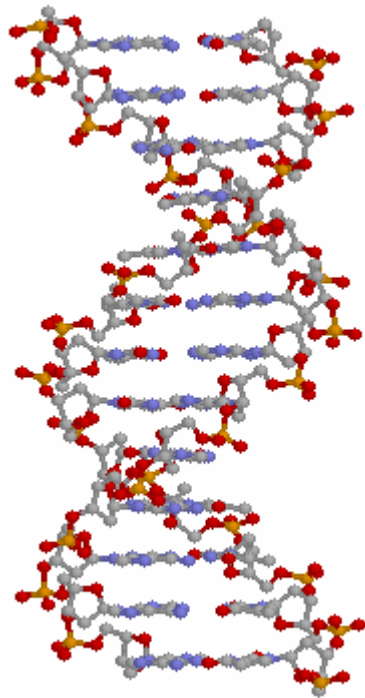
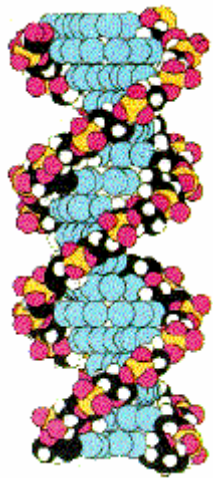
Algoritmos Genéticos

- ❑ **Locus** es la posición del gen en el cromosoma
- ❑ **Genoma** es el conjunto de material genético es decir todos los cromosomas
- ❑ **Genotipo** es un conjunto particular de genes en el genoma

■ Reproducción

- ❑ Es la **recombinación** o **cruzamiento** de los genes de los padres para formar un cromosoma totalmente nuevo
 - ❑ **Mutación** se refiere al hecho de que una descendencia puede cambiar en algo su ADN.
 - ❑ El **fitness** de un organismo es medido por el éxito del organismo en su vida
-

Algoritmos Genéticos



Algoritmos Genéticos

■ Historia

- Rechenberg (1960)
 - “Evolution strategies” (Evolutionsstrategie) - computación evolutiva
 - John Holland (1975)
 - “Adaption in Natural and Artificial Systems” algoritmos genéticos
 - John Koza (1992)
 - Inventó la programación genética (genetic programming) se usó Lisp
-

Algoritmos Genéticos

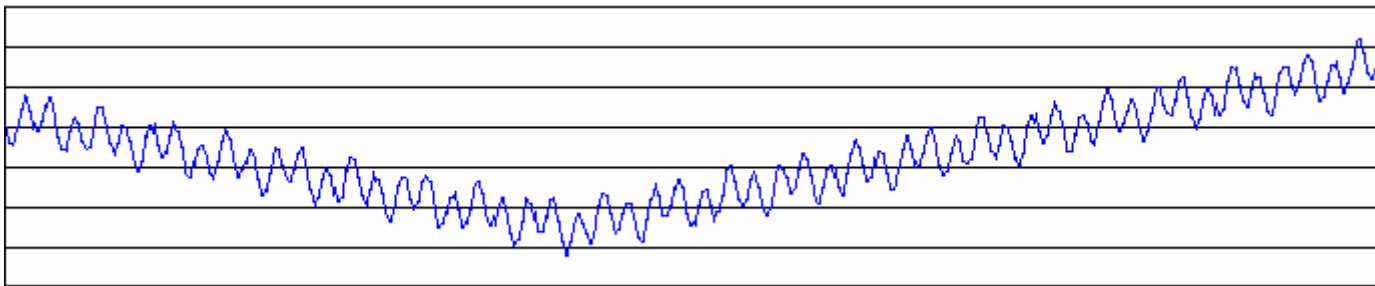
- Inteligencia artificial
 - Computación evolutiva
 - Algoritmos genéticos
 - Programación genética
 -

 - Inspirados en la teoría de la evolución de Darwin
-

Algoritmos Genéticos

■ Espacio de Búsqueda

- Espacio de todas las posibles soluciones, inicialmente poco conocido
- Más soluciones pueden ser generadas como el proceso de búsqueda continúe



Algoritmos Genéticos

- Problemas P vs NP
 - AG es una alternativa de solución para problemas NP
 - Lista de problemas de optimización NP

A compendium of NP optimization problems

<http://www.nada.kth.se/~viggo/problemlist/>

Algoritmos Genéticos

- Empieza con un conjunto de soluciones (representada mediante cromosomas) llamada población
 - Las soluciones son seleccionadas de acuerdo a su fitness, y forman nuevas soluciones (descendencia), mediante el cruzamiento y reproducción
-

Algoritmos Genéticos : Algoritmo general

1. [inicio] : Generar aleatoriamente n cromosomas (posibles soluciones para el problema)
 2. [fitness] : evaluar $f(x)$ para cada cromosoma x en la población
 3. [descendencia] : crear nueva poblacion
 1. [seleccion] : seleccionar cromosomas de acuerdo a $f(x)$ (mayor fitness mayor oportunidad)
 2. [cruzamiento] : cruzar cromosomas con alguna probabilidad de cruzamiento, si no la descendencia es la copia exacta de los padres
 1. [mutación] : Con probabilidad de mutación mutar la descendencia en su locus
 2. [aceptación] : localizar la nueva descendencia como nueva población
 4. [reemplazo] : Usar la descendencia para correr el algoritmo
 5. [test] : si se satisface alguna condicion parar, y retornar la mejor solucion del problema
 6. [Loop] : retornar al paso 2
-

Algoritmos Genéticos: Operadores

■ Codificación de cromosomas

- Contienen información de la solución que representan, la manera más usada de codificar es usando cadenas binarias

- Cromosoma 1 1101100100110110

- Cromosoma 2 1101111000011110

Holland lo llama Genotipo mientras Schafer

Lo llama cromosoma

Algoritmos Genéticos: Operadores

- Funcion de evaluacion y fitness
 - Función de evaluación o función objetivo da una medida de performance con respecto a un conjunto de parámetros, es independiente entre cromosomas
 - Función fitness transforma esa medida de performance en oportunidades de reproducción, y está definida con respecto a otros miembros de la población

$$\text{fitness} = f_i / f_{\text{promedio}}$$

f_i = funcion de evaluacion de cada cromosoma

f_{promedio} = promedio de la funcion de evaluación de todas los cromosomas en la población

Algoritmos Genéticos: Operadores

■ Cruzamiento

- ❑ Escoger aleatoriamente algún punto (o varios depende del problema) de cruce, y copiar la información de un cromosoma hacia otro a partir del punto generado

- ❑ **Cromosona 1** 11011 | 00100110110

- ❑ **Cromosona 2** 11011 | 11000011110

- ❑ **Descendencia 1** 11011 | 11000011110

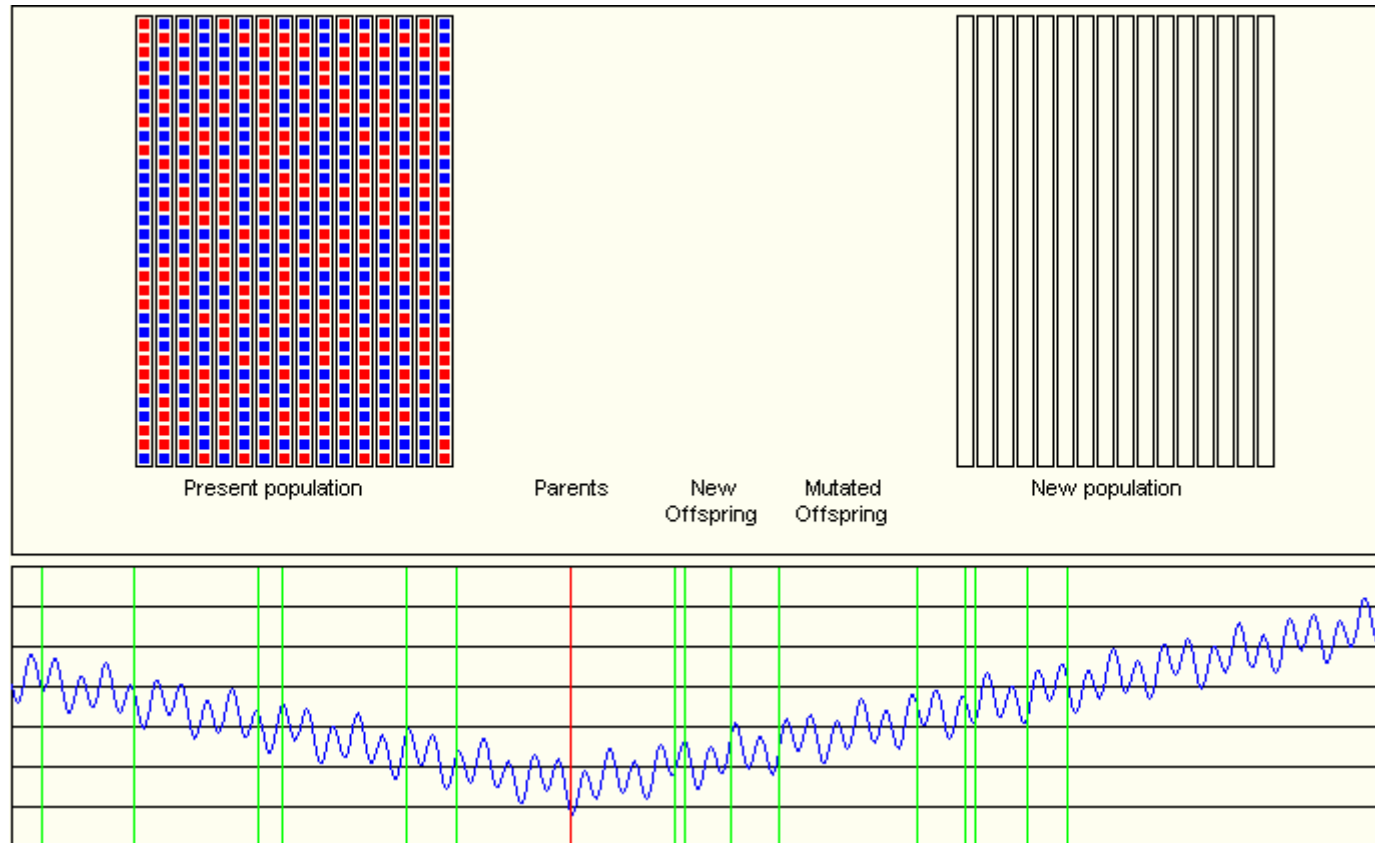
- ❑ **Descendencia 2** 11011 | 00100110110

Algoritmos Genéticos: Operadores

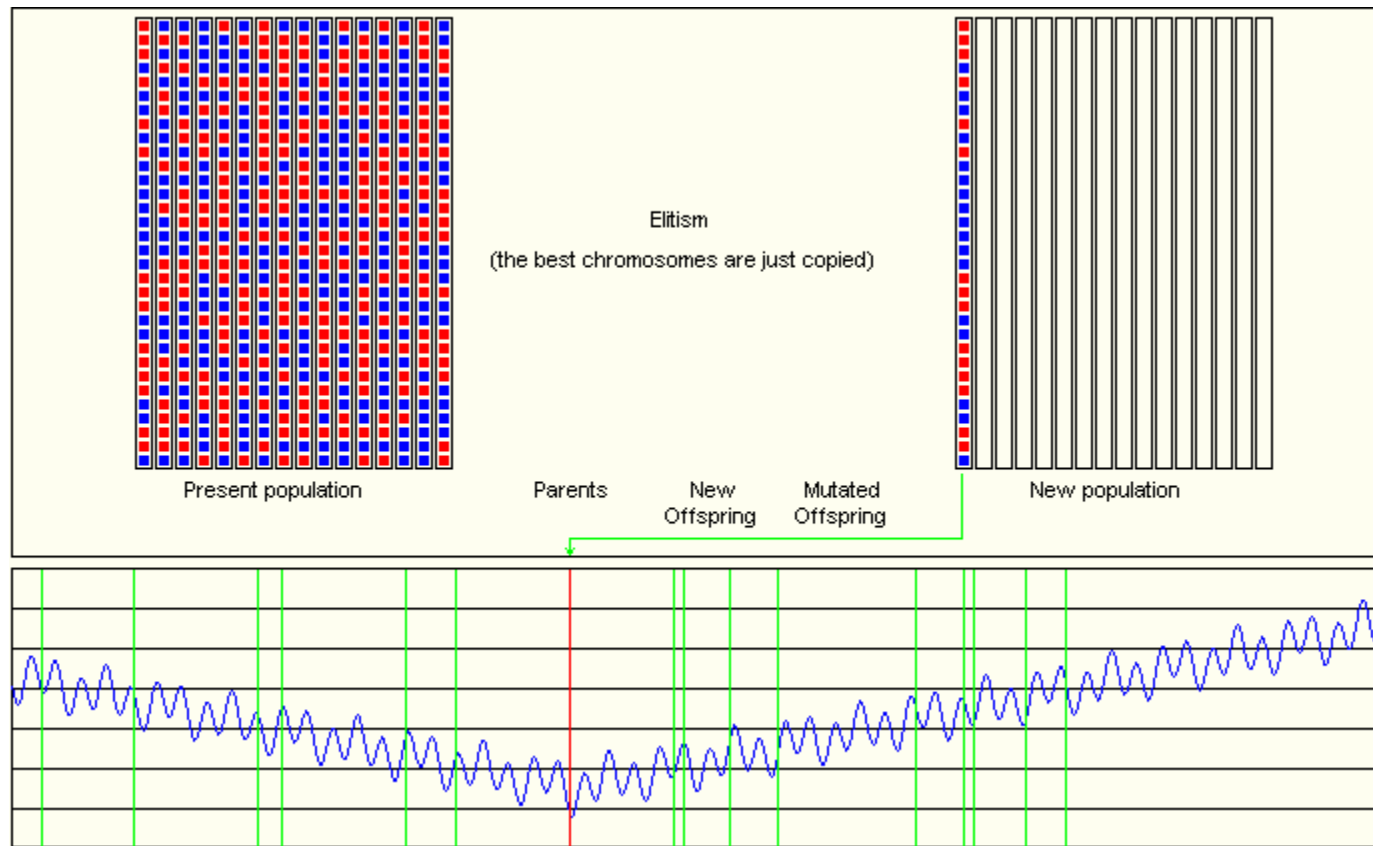
■ Mutación

- ❑ Útil para salir o evitar mínimos locales
 - ❑ Cambia aleatoriamente la descendencia
 - ❑ Original descendencia 1 110**1**111000011110
 - ❑ Original descendencia 2 110110**0**100110110
 - ❑ Descendencia mutada 1 110**0**111000011110
 - ❑ Descendencia mutada 2 110110**1**100110110
-

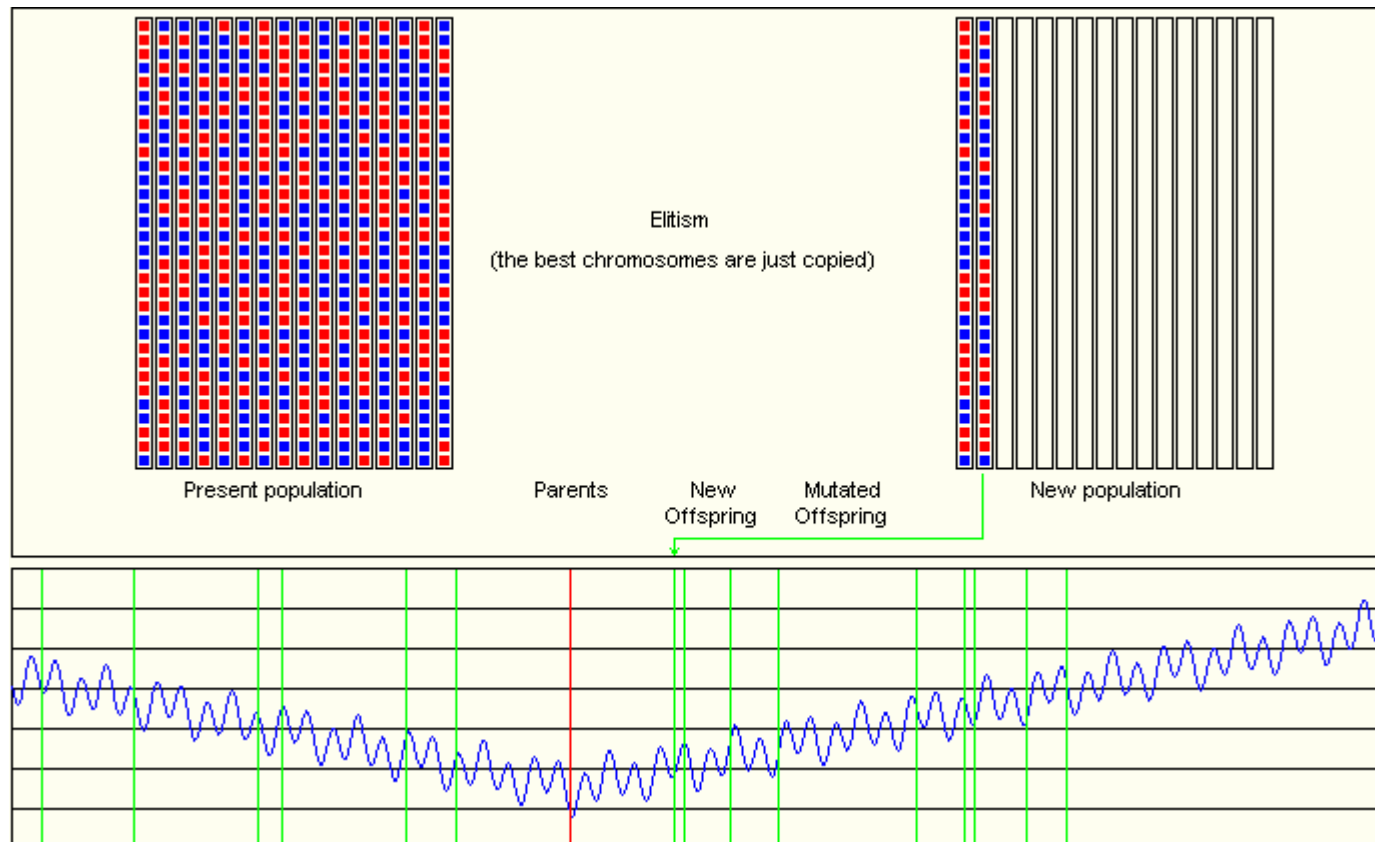
Algoritmos Genéticos: Ejemplo



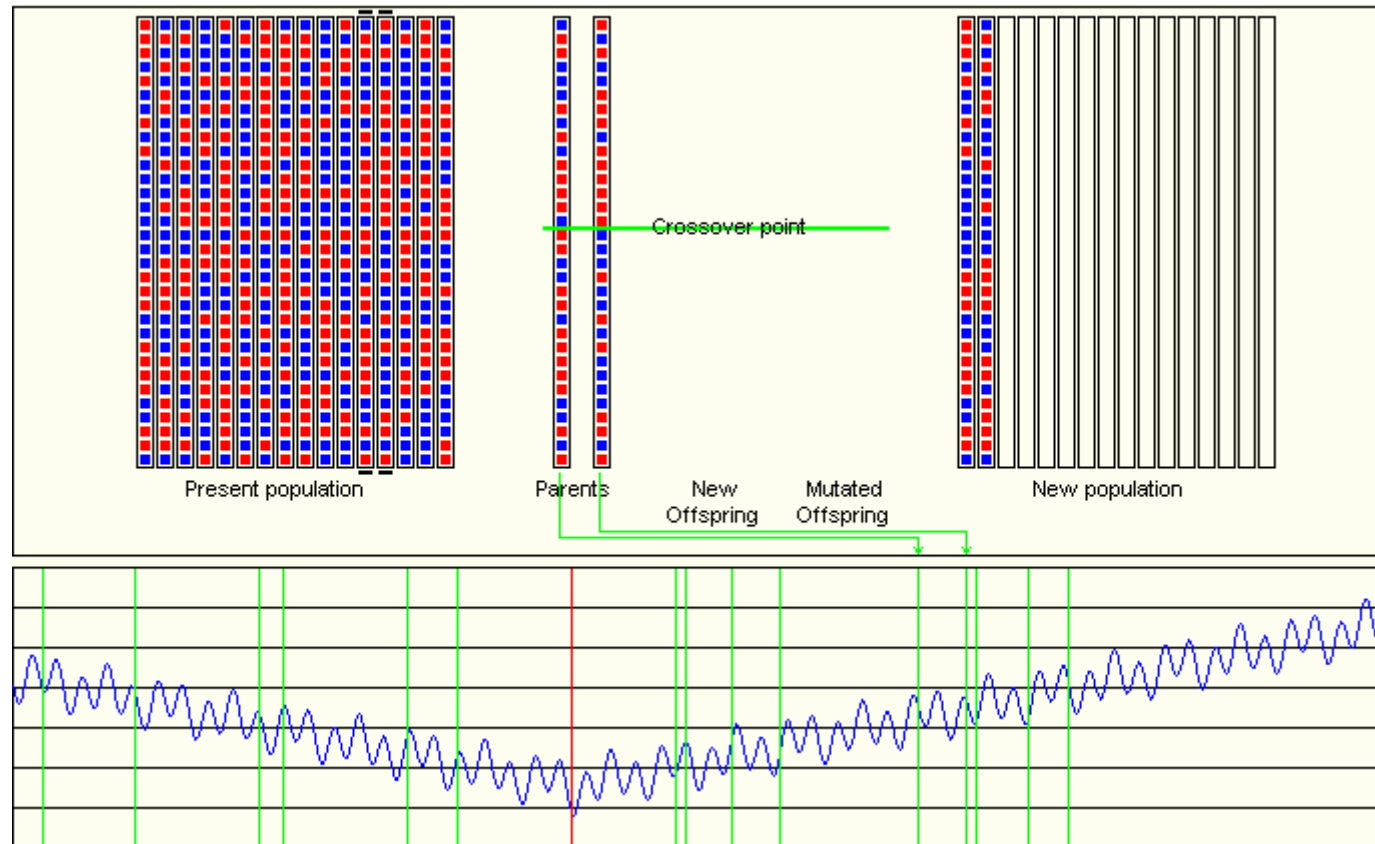
Algoritmos Genéticos: Ejemplo



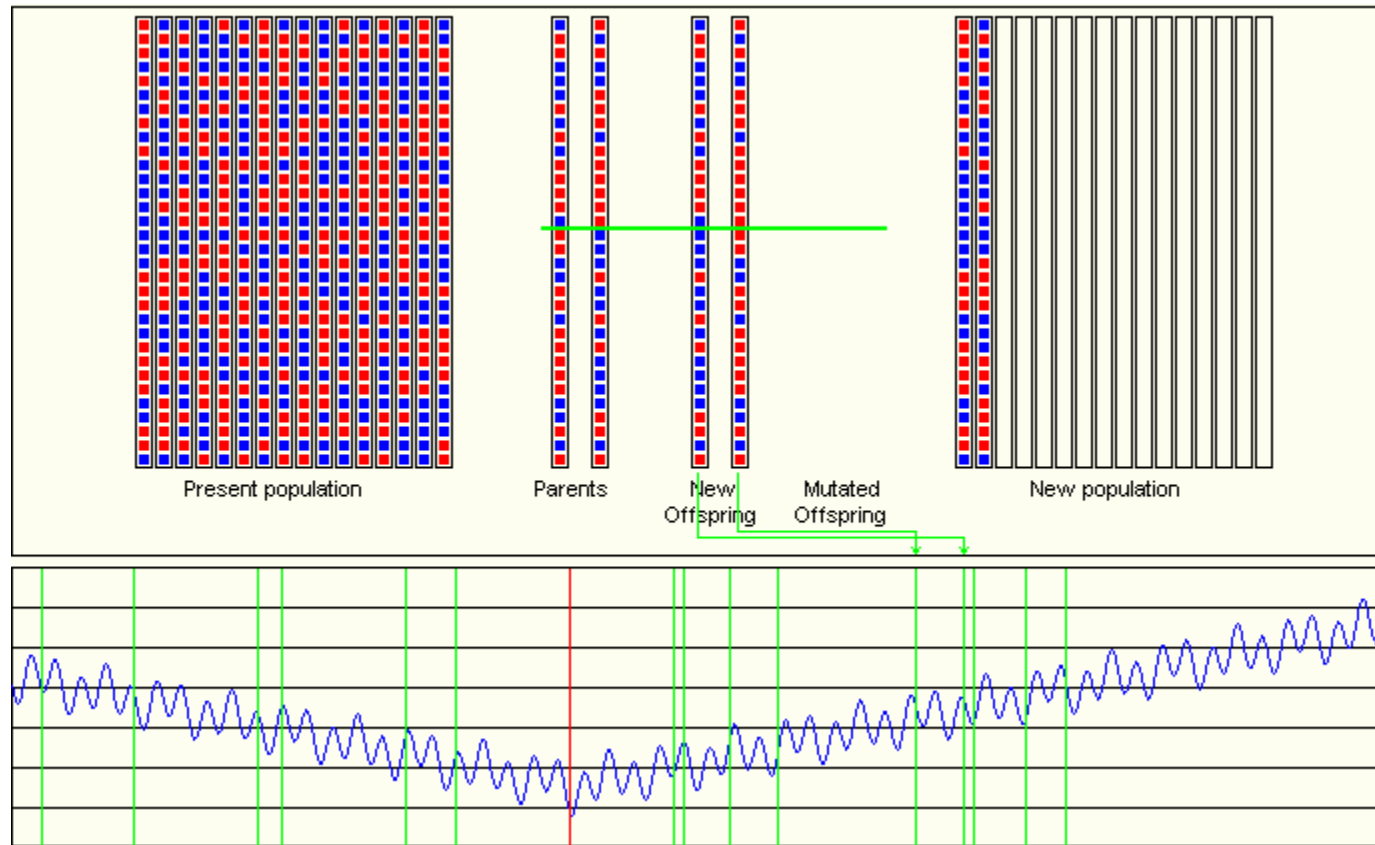
Algoritmos Genéticos: Ejemplo



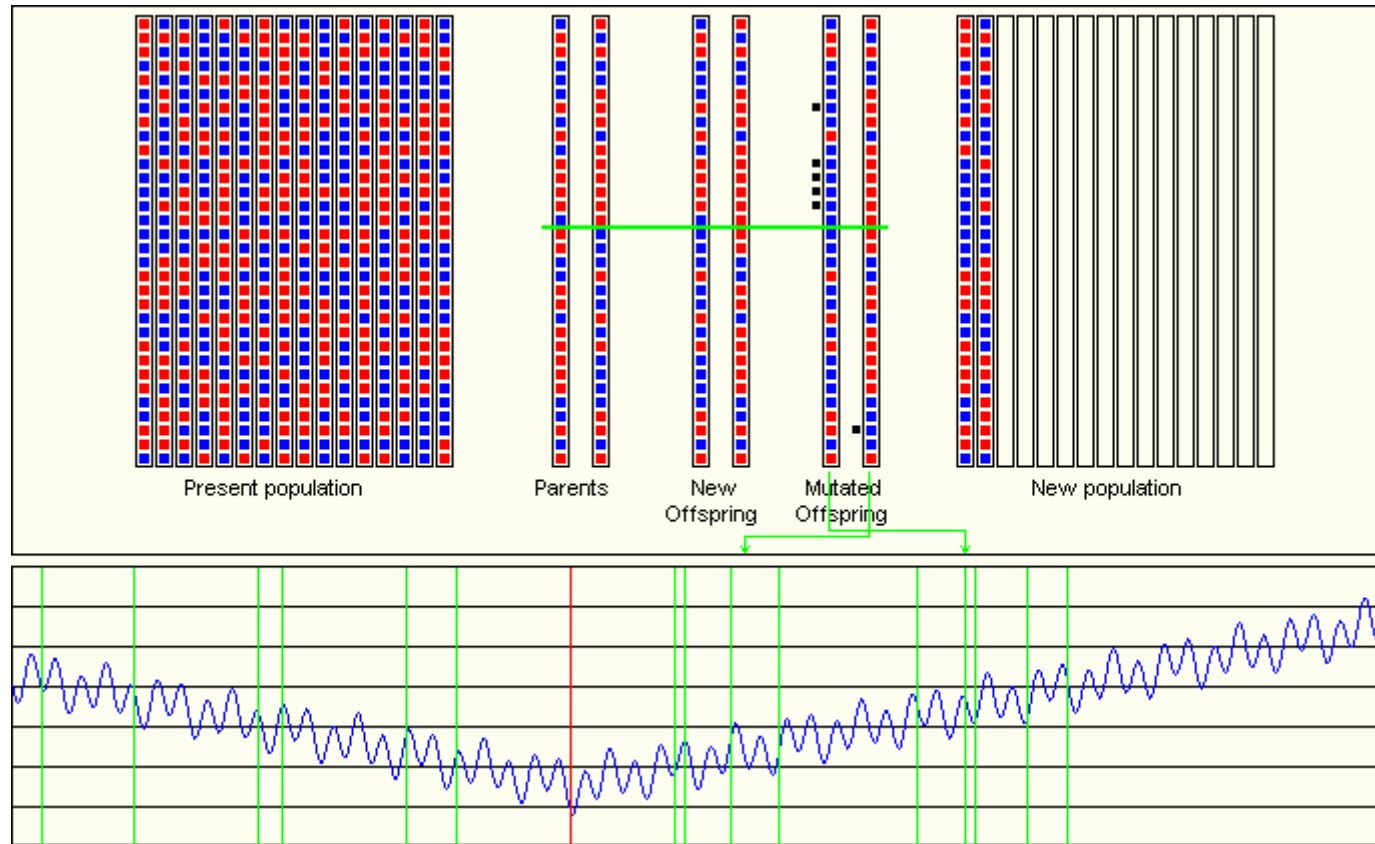
Algoritmos Genéticos: Ejemplo



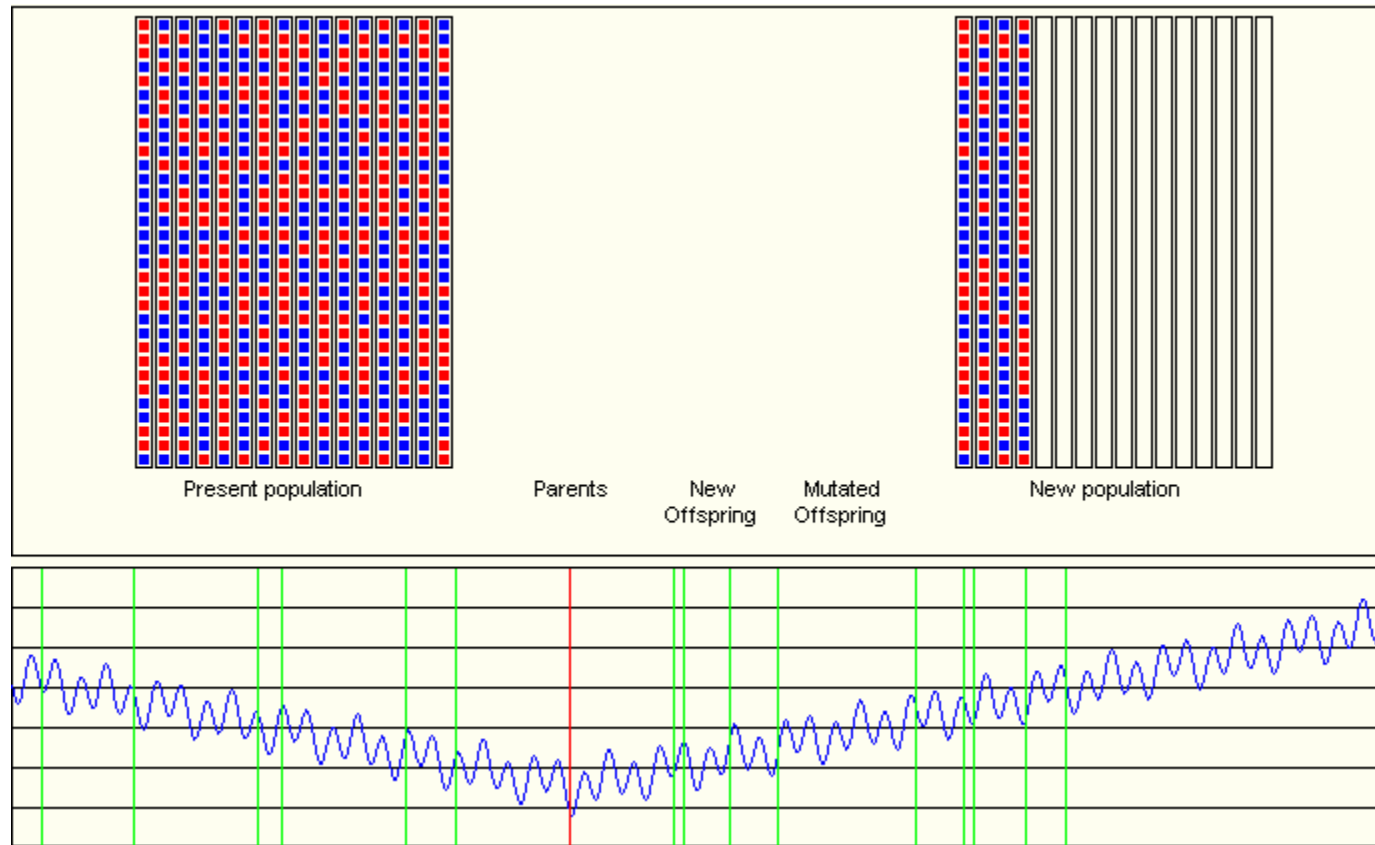
Algoritmos Genéticos: Ejemplo



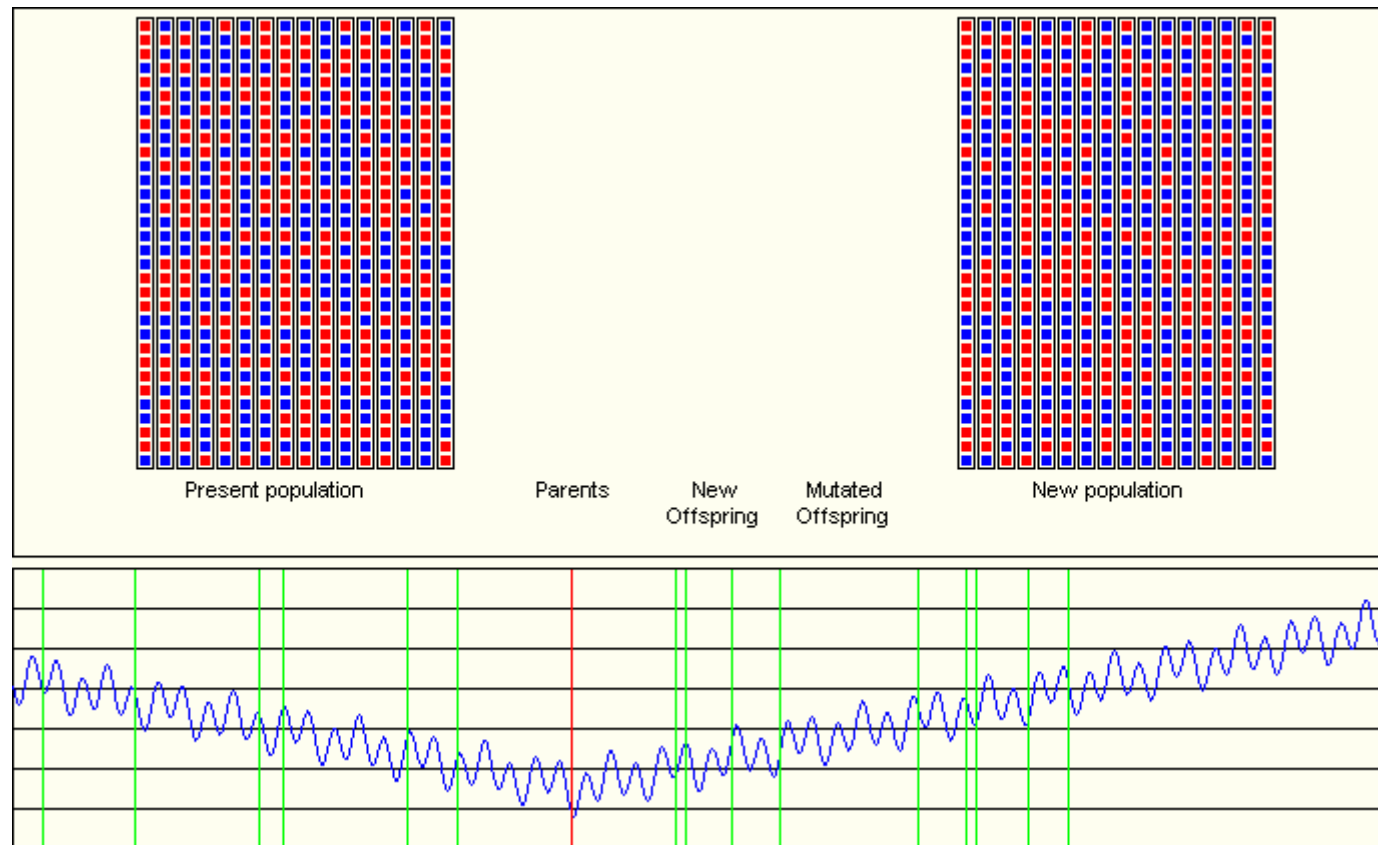
Algoritmos Genéticos: Ejemplo



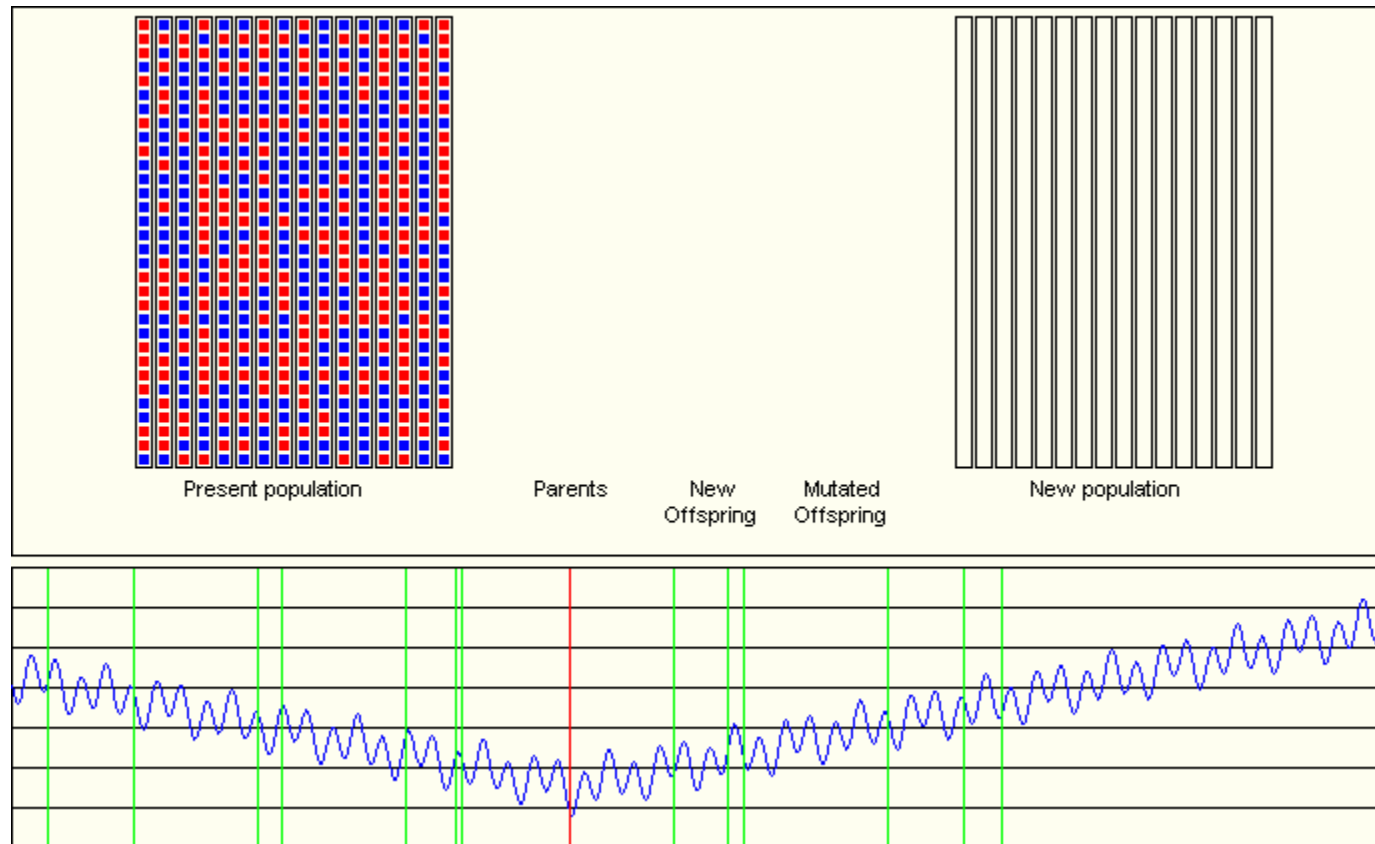
Algoritmos Genéticos: Ejemplo



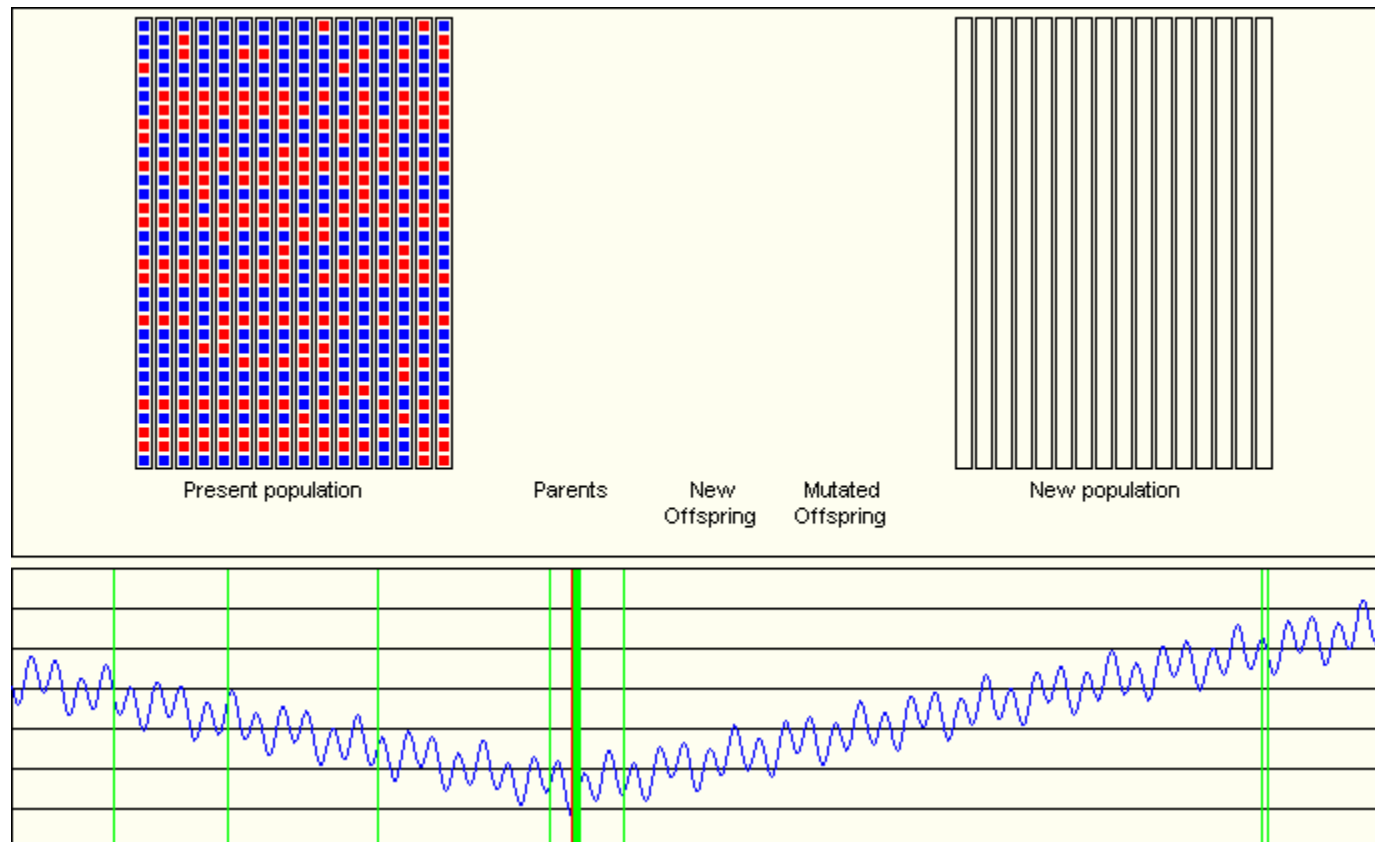
Algoritmos Genéticos: Ejemplo



Algoritmos Genéticos: Ejemplo



Algoritmos Genéticos: Ejemplo



Algoritmos Genéticos: Parámetros

■ Probabilidad de cruce

- 100% todos los descendientes son hechos por cruzamientos
- 0% no se usa cruzamiento

■ Probabilidad de mutación

- No ocurre mucho pues si esto pasa se estaría realizando una búsqueda aleatoria
-

Algoritmos Genéticos: Parámetros

■ Tamaño de la Población

- Cuantos cromosomas existen en la población de una generación, si es muy pequeña el espacio de búsqueda es limitado, si es muy grande el algoritmo se vuelve muy lento
-

Algoritmos Genéticos: Parámetros

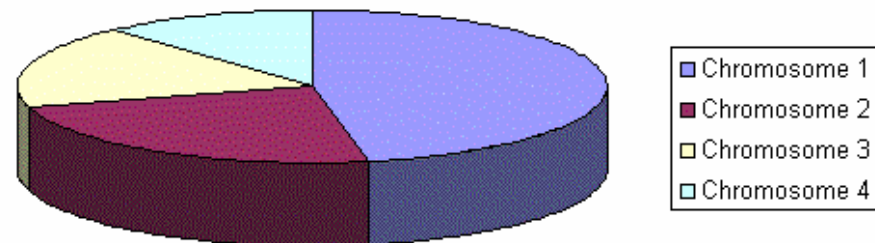
■ Selección

- La teoría de la evolución de Darwin sugiere que solamente los mejores deben sobrevivir y formar una nueva población
 - Algunas técnicas computacionales para seleccionar los mejores cromosomas son las siguientes:
 - Selección de la ruleta
 - Selección de Boltzman
 - Selección mediante torneo
 - Selección de ranking
 - Selección de estado de equilibrio
-

Algoritmos Genéticos: Parámetros

■ Selección

□ Selección de la ruleta



- 1 **[sumar]** : S es la suma de todos los fitness de los cromosomas en una población
 - 2 **[seleccionar]** : Generar un número aleatorio r en el rango $(0, S)$
 - 3 **[Loop]** : sumar los fitness de la población de 0 a S , si $S > r$ parar y retornar cromosoma
-

Algoritmos Genéticos: Parámetros

■ Selección

- Selección de estado de equilibrio
 - La idea es que la mayor parte de los cromosomas sobrevivan a la siguiente generación
 - Unos pocos cromosomas con alto fitness se reproducen, esta descendencia reemplazará a algunos cromosomas con bajo fitness que serán eliminados
 - El resto de cromosomas sobrevive a la siguiente generación
-

Algoritmos Genéticos: Parámetros

■ Selección

□ Elitismo

- Se selecciona el mejor cromosoma, (o algunos) y sobreviven a la siguiente generación, el resto de cromosomas son sometidos a cruzamiento
 - Generalmente incrementa la performance de los algoritmos genéticos pues de cierta manera evita perder buenas soluciones
-

Algoritmos Genéticos: Parámetros

■ Codificación

- Depende mucho del problema

- Codificación Binaria

- Cromosoma A

101100101100101011100101

- Cromosoma B

111111100000110000011111

Ejemplo problema de la mochila

Algoritmos Genéticos: Parámetros

■ Codificación

□ Codificación por Permutación

□ Cromosomas son cadenas de números, que representan una secuencia

□ Cromosoma A 1 5 3 2 6 4 7 9 8

□ Cromosoma B 8 5 6 7 2 3 1 4 9

□ Ejemplo TSP , cada cromosoma lleva la lista ordenada de ciudades que el agente visitará

Algoritmos Genéticos: Parámetros

■ Codificación

□ Codificación por Valor

- Cada cromosoma es una cadena de valores, los valores pueden ser números enteros, reales, caracteres, etc

Cromosoma A 1.2324 5.3243 0.4556 2.3293 2.4545

Cromosoma B ABDJEIFJDHDIERJFDLDFLFEGT

Cromosoma C (back), (back), (right), (forward), (left)

Ejemplo: Buscar Pesos para una red neuronal

Algoritmos Genéticos: Parámetros

■ Codificación

□ Codificación de Árbol

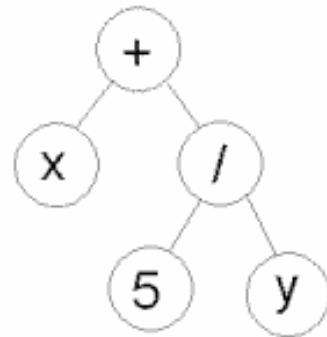
- Usado en la programación genética, involucra programas o expresiones
 - Cada cromosoma es un árbol de algunos objetos, como funciones, comandos etc
-

Algoritmos Genéticos: Parámetros

■ Codificación

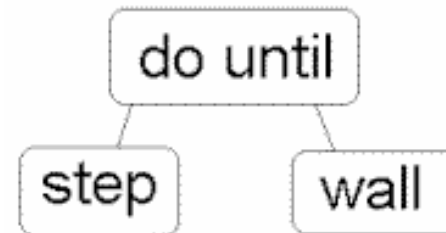
- Codificación de Árbol Usualmente se usa LISP

Cromosoma A



(+ x (/ 5 y))

Cromosoma B



(do_until step wall)

Algoritmos Genéticos: Parámetros

■ Cruzamiento y Mutación (binaria)

□ Cruzamiento de un punto



$$11001011 + 11011111 = 11001111$$

□ Cruzamiento de dos puntos



$$11001011 + 11011111 = 11011111$$

Algoritmos Genéticos: Parámetros

■ Cruzamiento y Mutación (binaria)

- Cruzamiento uniforme, los bits son copiados aleatoriamente



$$11001011 + 11011101 = 11011111$$

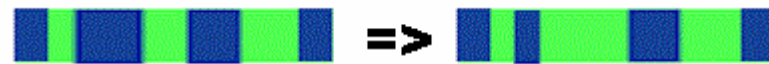
- Cruzamiento aritmético, se realiza alguna operación aritmética



$$11001011 + 11011111 = 11001001 \text{ (AND)}$$

Algoritmos Genéticos: Parámetros

- Cruzamiento y Mutación (binaria)
 - Mutación Inversión de bits



11001001 => 10001001

Algoritmos Genéticos: Parámetros

- Cruzamiento y Mutación (permutación)

- Cruzamiento de un punto

- $(1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9) + (4\ 5\ 3\ 6\ 8\ 9\ 7\ 2\ 1) = (1\ 2\ 3\ 4\ 5\ 6\ 8\ 9\ 7)$

- Mutación : Order Chaining

- Dos números son seleccionados y cambiados

- $(1\ 2\ 3\ 4\ 5\ 6\ 8\ 9\ 7) \Rightarrow (1\ 8\ 3\ 4\ 5\ 6\ 2\ 9\ 7)$



Algoritmos Genéticos: Parámetros

- Cruzamiento y Mutación (valor)
 - Cruzamiento
 - De la misma manera que en la forma binaria
 - Mutación : Order Chaining
 - En el caso de reales sumar o sustraer pequeños valores

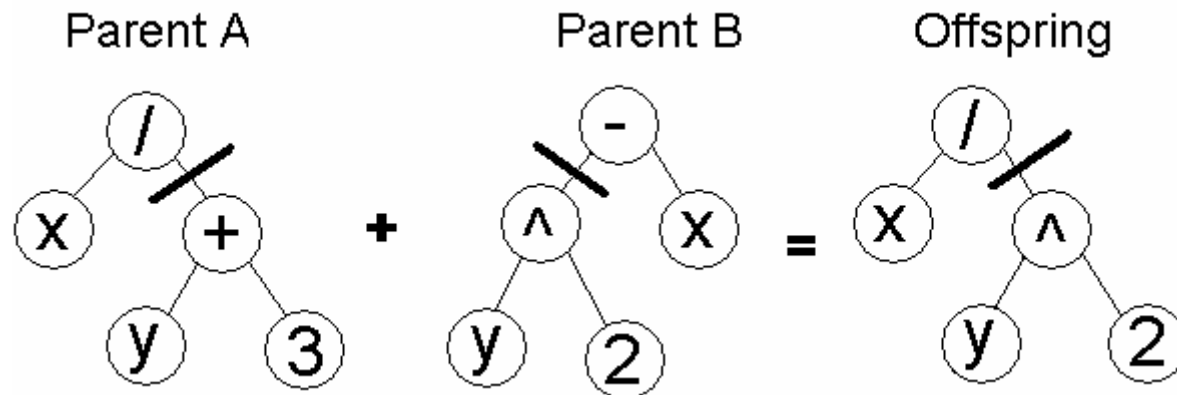
(1.29 5.68 2.86 4.11 5.55) => (1.29 5.68 2.73 4.22 5.55)

Algoritmos Genéticos: Parámetros

■ Cruzamiento y Mutación (árbol)

□ Cruzamiento de árbol

Se selecciona en ámbos padres un punto de cruce y se intercambian los subárboles

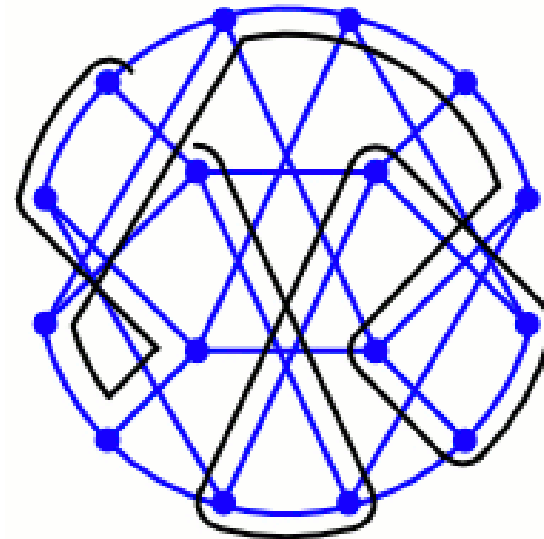
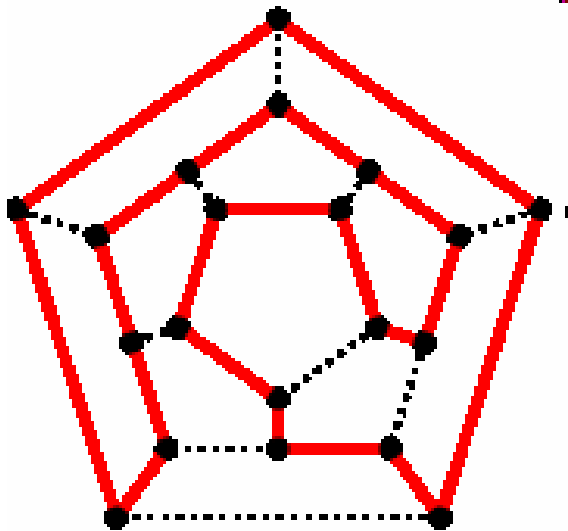


Mutación : Se selecciona un nodo que será cambiado

Algoritmos Genéticos: Ejemplo TSP

- Buscar un tour Hamiltoniano minimal en un grafo completo de N nodos, (existe tambien el TSP para grafos no completos)
 - Camino Hamiltoniano:
 - Es un grafo no dirigido que visita cada vértice solamente una vez
 - Ciclo Hamiltoniano:
 - Si a la vez es un camino hamiltoniano y retorna al vértice inicial
 - Problema del camino Hamiltoniano
 - Determinar si existen caminos y ciclos hamiltonianos es un problema del tipo NP
-

Algoritmos Genéticos: Ejemplo TSP



Algoritmos Genéticos: Ejemplo TSP

■ Implementación

- Población
 - 16 cromosomas codificados con permutación
 - Cruzamiento
 - Cruzamiento de un punto:
 - Parte del primer padre es copiado en el segundo padre, y el resto mantiene el mismo orden
 - Cruzamiento de dos puntos
 - Dos partes del primer padre son copiados y el resto se mantienen en el mismo orden del segundo padre
 - Mutación
 - Se intercambian algunas ciudades de manera aleatoria
-

Algoritmos Genéticos: Recomendaciones

■ Cruzamiento

- Tasa debe ser alta $> 80\%$, sin embargo en algunos problemas puede ser menor

■ Mutacion

- Debe ser baja $< 1\%$

■ Poblacion

- Muy grande no incrementa la performance
- 20 a 30, sin embargo el tamaño de la poblacion depende de la codificacion

■ Selección

- Puede utilizarse cualquier método descrito anteriormente

■ Codificacion

- Depende del problema
-

Algoritmos Genéticos: Aplicaciones

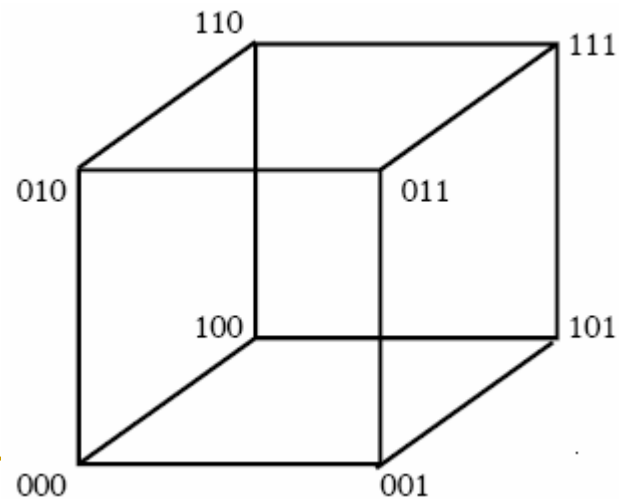
- Problemas NP, arte, musica
 - Sistemas lineales no dinámicos - predicción, análisis de datos
 - Diseño de redes neuronales, arquitectura y pesos
 - Trayectoria de Robots
 - Programas LISP evolutivos (programacion genética)
 - Estrategias de planificación
 - Encontrar la forma de moléculas de proteínas
 - TSP y secuencias de programación
 - Funciones para crear imágenes
-

Algoritmos genéticos: espacios de búsqueda como hipercubos

- ¿Por qué trabajan los algoritmos genéticos?
 - Holland dio algunos argumentos para explicar como un algoritmo genético podía resultar en una búsqueda compleja y robusta haciendo muestreos de manera implícita, en las particiones de un hiperplano en un espacio de búsqueda
-

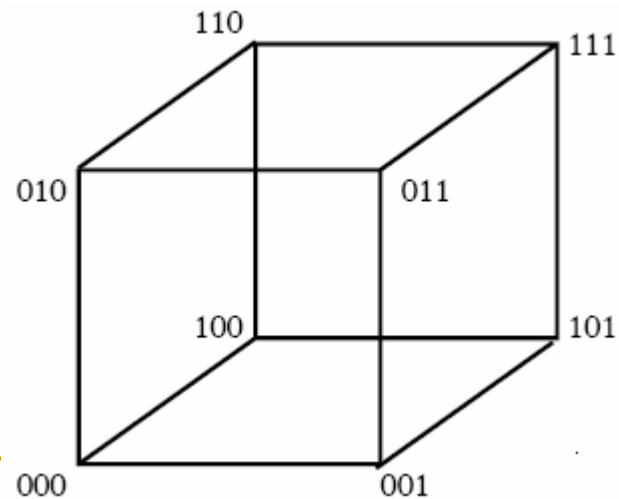
Algoritmos genéticos: espacios de búsqueda como hipercubos

- Espacio generado por cromosomas codificados (binaria) con 3 bits,
- cada bit adyacente difiere en un bit
- 0** representa el plano frontal



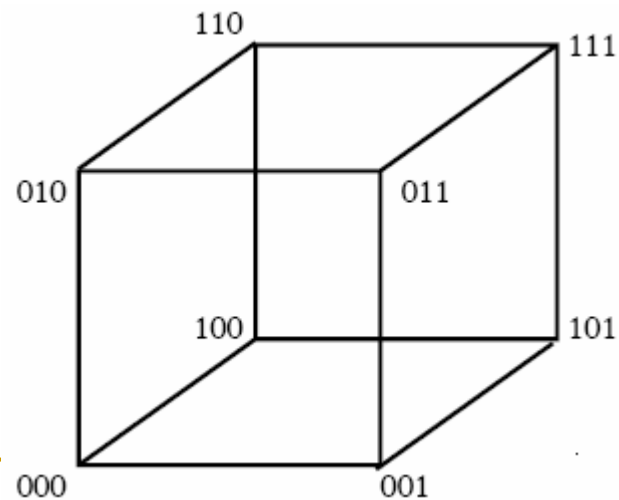
Algoritmos genéticos: espacios de búsqueda como hipercubos

- Las cadenas (cromosomas) que contienen * se les llamará **esquemas** (schemata)
- Cada esquema corresponde a un **hiperplano** en el espacio de búsqueda

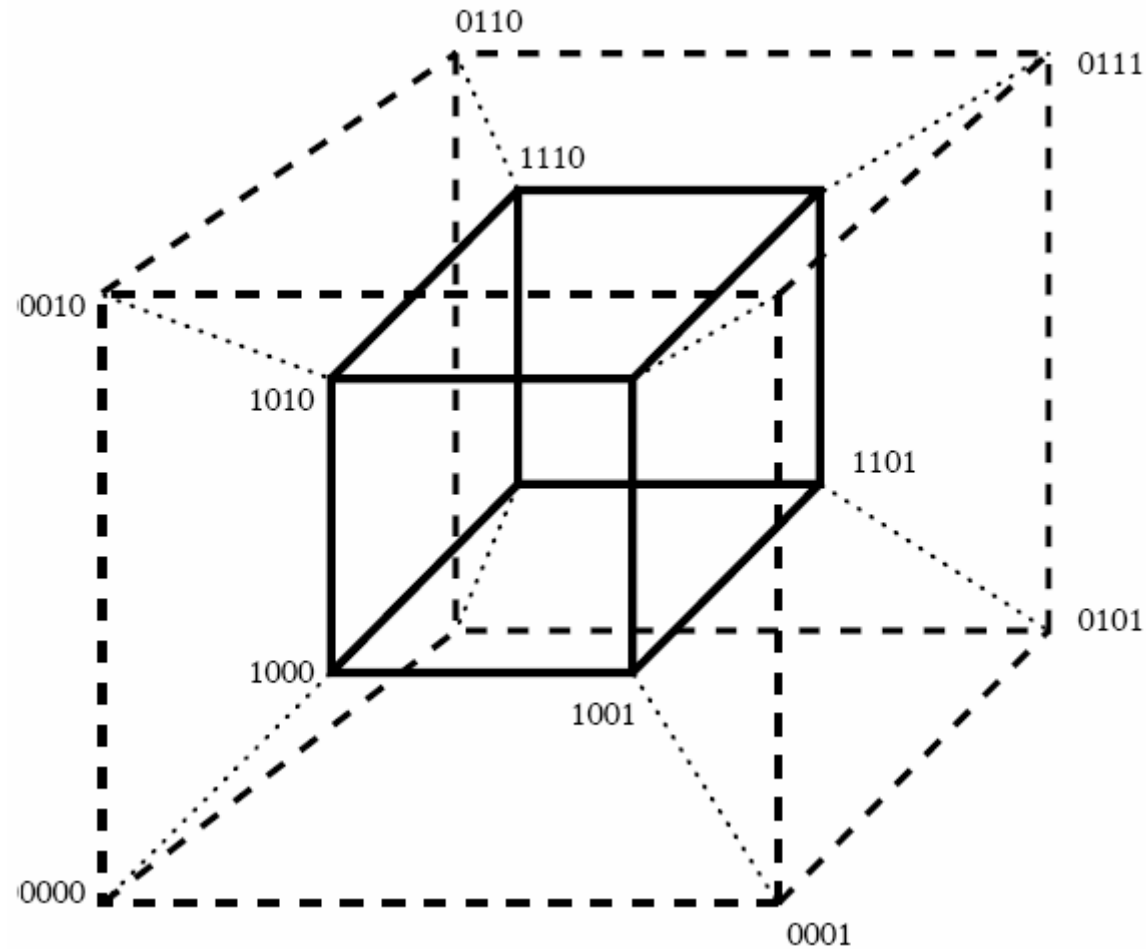


Algoritmos genéticos: espacios de búsqueda como hipercubos

- El **orden** de un hiperplano son el número de bits que aparecen en determinado esquema $1^{**}1^{*****}0^{**}$ es un hiperplano de orden 3
- Cada cromosoma es miembro de 2^L-1 hiperplanos $L=$ longitud del cromosoma
- Sobre todo el espacio de búsqueda se pueden definir 3^L-1 hiperplanos



Algoritmos genéticos: espacios de búsqueda como hipercubos



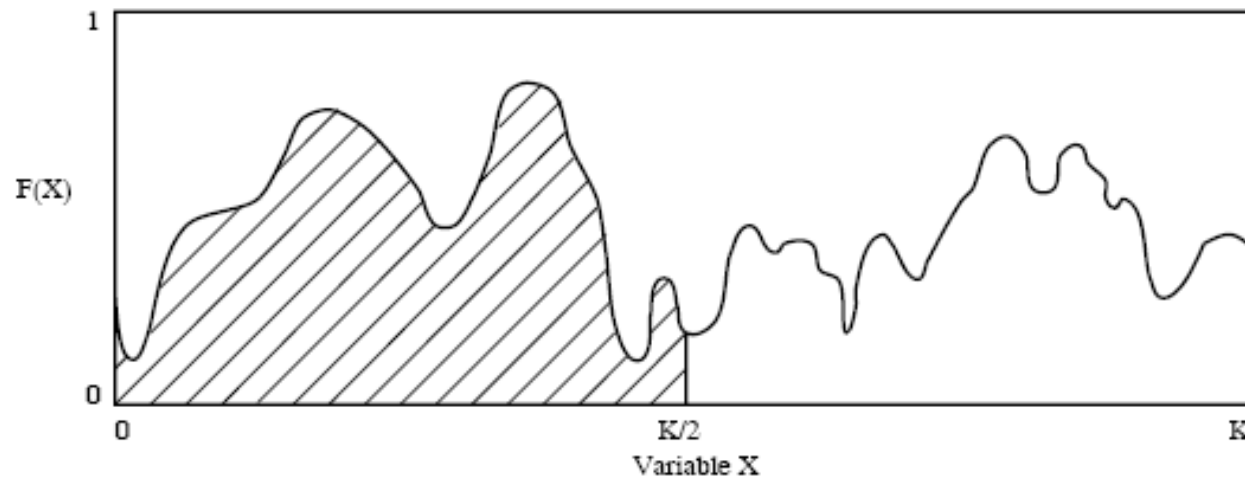
Algoritmos genéticos: espacios de búsqueda como hipercubos

- Establecer que cada cadena es miembro de $2^L - 1$ hiperplanos, no da mucha información si un determinado punto es examinado aisladamente en el espacio de búsqueda
 - La idea de **búsqueda basada en población** es una parte crucial en los algoritmos genéticos
 - La población da información de **numerosos hiperplanos**
 - La idea principal es que muchos hiperplanos son evaluados de una manera implícita de forma paralela cada vez que cada cadena es evaluada (Holland)
 - El efecto acumulativo de evaluar la población da información estadística de los hiperplanos
-

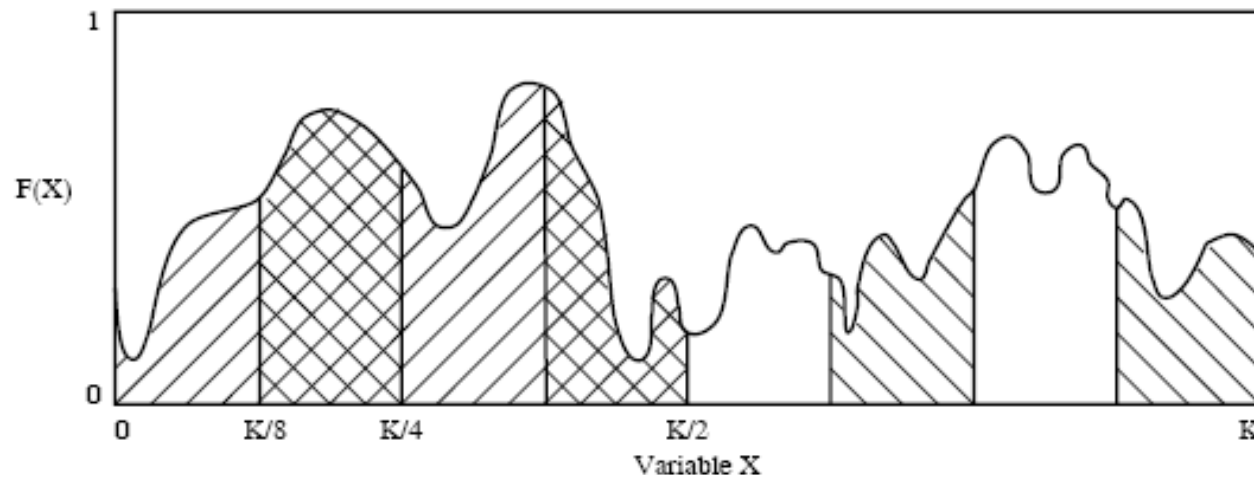
Algoritmos genéticos: espacios de búsqueda otra manera de ver

- Se puede ver como una función sobre una simple variable
 - El hiperplano $0^{*****} \dots^{**}$ actúa sobre la primera mitad y $1^{***} \dots^{**}$ actual en la otra mitad
-

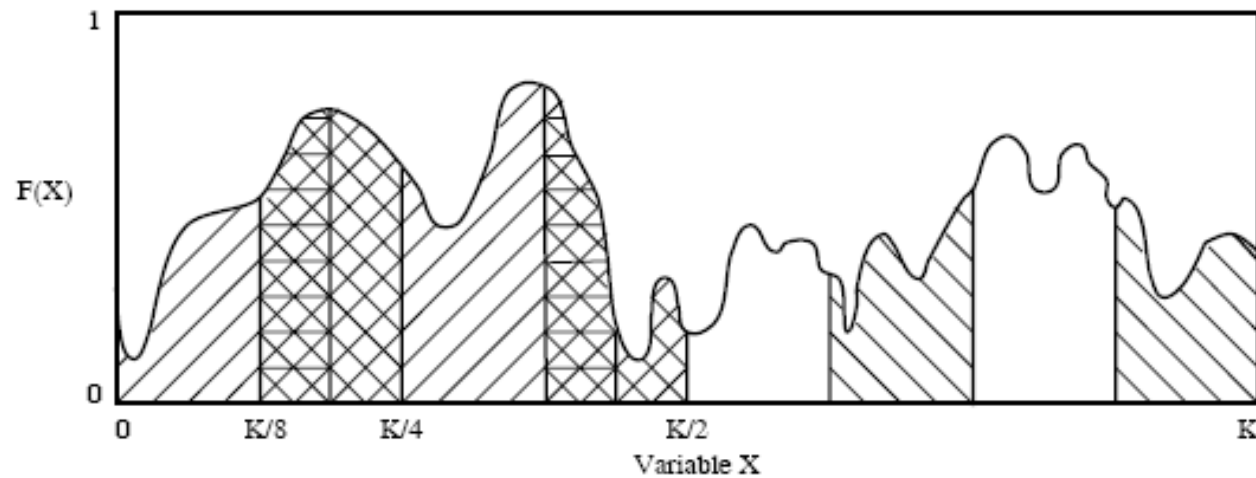
Algoritmos genéticos: espacios de búsqueda otra manera de ver



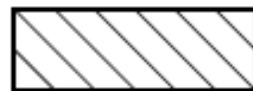
Algoritmos genéticos: espacios de búsqueda otra manera de ver



Algoritmos genéticos: espacios de búsqueda otra manera de ver



0***...*



**1*...*



0*10*...*

Algoritmos genéticos: espacios de búsqueda otra manera de ver

| String | Fitness | Random | Copies | | String | Fitness | Random | Copies |
|-------------------------------|---------|--------|--------|--|-------------------------------|---------|--------|--------|
| 001 $b_{1,4} \dots b_{1,L}$ | 2.0 | – | 2 | | 011 $b_{12,4} \dots b_{12,L}$ | 0.9 | 0.28 | 1 |
| 101 $b_{2,4} \dots b_{2,L}$ | 1.9 | 0.93 | 2 | | 000 $b_{13,4} \dots b_{13,L}$ | 0.8 | 0.13 | 0 |
| 111 $b_{3,4} \dots b_{3,L}$ | 1.8 | 0.65 | 2 | | 110 $b_{14,4} \dots b_{14,L}$ | 0.7 | 0.70 | 1 |
| 010 $b_{4,4} \dots b_{4,L}$ | 1.7 | 0.02 | 1 | | 110 $b_{15,4} \dots b_{15,L}$ | 0.6 | 0.80 | 1 |
| 111 $b_{5,4} \dots b_{5,L}$ | 1.6 | 0.51 | 2 | | 100 $b_{16,4} \dots b_{16,L}$ | 0.5 | 0.51 | 1 |
| 101 $b_{6,4} \dots b_{6,L}$ | 1.5 | 0.20 | 1 | | 011 $b_{17,4} \dots b_{17,L}$ | 0.4 | 0.76 | 1 |
| 011 $b_{7,4} \dots b_{7,L}$ | 1.4 | 0.93 | 2 | | 000 $b_{18,4} \dots b_{18,L}$ | 0.3 | 0.45 | 0 |
| 001 $b_{8,4} \dots b_{8,L}$ | 1.3 | 0.20 | 1 | | 001 $b_{19,4} \dots b_{19,L}$ | 0.2 | 0.61 | 0 |
| 000 $b_{9,4} \dots b_{9,L}$ | 1.2 | 0.37 | 1 | | 100 $b_{20,4} \dots b_{20,L}$ | 0.1 | 0.07 | 0 |
| 100 $b_{10,4} \dots b_{10,L}$ | 1.1 | 0.79 | 1 | | 010 $b_{21,4} \dots b_{21,L}$ | 0.0 | – | 0 |
| 010 $b_{11,4} \dots b_{11,L}$ | 1.0 | – | 1 | | | | | |

Algoritmos genéticos: espacios de búsqueda otra manera de ver

| Schemata and Fitness Values | | | | | | | | | | |
|-----------------------------|-------|-------|--------|-----|--|----------|-------|-------|--------|-----|
| Schema | Mean | Count | Expect | Obs | | Schema | Mean | Count | Expect | Obs |
| 101*...* | 1.70 | 2 | 3.4 | 3 | | *0**...* | 0.991 | 11 | 10.9 | 9 |
| 111*...* | 1.70 | 2 | 3.4 | 4 | | 00**...* | 0.967 | 6 | 5.8 | 4 |
| 1*1*...* | 1.70 | 4 | 6.8 | 7 | | 0***...* | 0.933 | 12 | 11.2 | 10 |
| *01*...* | 1.38 | 5 | 6.9 | 6 | | 011*...* | 0.900 | 3 | 2.7 | 4 |
| **1*...* | 1.30 | 10 | 13.0 | 14 | | 010*...* | 0.900 | 3 | 2.7 | 2 |
| *11*...* | 1.22 | 5 | 6.1 | 8 | | 01**...* | 0.900 | 6 | 5.4 | 6 |
| 11**...* | 1.175 | 4 | 4.7 | 6 | | 0*0*...* | 0.833 | 6 | 5.0 | 3 |
| 001*...* | 1.166 | 3 | 3.5 | 3 | | *10*...* | 0.800 | 5 | 4.0 | 4 |
| 1***...* | 1.089 | 9 | 9.8 | 11 | | 000*...* | 0.767 | 3 | 2.3 | 1 |
| 0*1*...* | 1.033 | 6 | 6.2 | 7 | | **0*...* | 0.727 | 11 | 8.0 | 7 |
| 10**...* | 1.020 | 5 | 5.1 | 5 | | *00*...* | 0.667 | 6 | 4.0 | 3 |
| *1**...* | 1.010 | 10 | 10.1 | 12 | | 110*...* | 0.650 | 2 | 1.3 | 2 |
| ****...* | 1.000 | 21 | 21.0 | 21 | | 1*0*...* | 0.600 | 5 | 3.0 | 4 |
| | | | | | | 100*...* | 0.566 | 3 | 1.70 | 2 |

Teorema de Esquemas de Holland (Schema Theorem)

- Holland en su trabajo muestra el teorema de Esquemas, como una manera de proveer un modelo formal de efectividad de los algoritmos genéticos en el proceso de búsqueda

[1] J.H. Holland, (1998) *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to biology, control and artificial intelligence*. MIT Press, ISBN 0-262-58111-6. (NB original printing 1975).

Teorema de Esquemas de Holland (Schema Theorem)

■ Definición 1 Esquema H

- Un esquema es un subconjunto del espacio de todos los posibles individuos para los cuales todos los genes hacen matching con la plantilla H

- Ejemplo: el esquema H identifica el conjunto de cromosomas

- $H = [0\ 1\ *\ 1\ *]$

| | | | | |
|---|---|---|---|---|
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 |



Teorema de Esquemas de Holland (Schema Theorem)

■ Definición 2 Orden del esquema $o(H)$

- Es el número de genes diferentes de * en un esquema en particular
- Ejemplo $o(* * * 0 * * *) = 1$

■ Definición 3 Longitud del esquema $\delta(H)$.

- Es la distancia entre el primero y el último gen diferente de *
 - Ejemplo $\delta(* * * 0 * * *) = 4 - 4 = 0$
-

Teorema de Esquemas de Holland (Schema Theorem)

■ Operadores de selección - selección proporcional a fitness

- $P(h \in H)$
 - ¿Cuál es la probabilidad de h en H?
- $m(H,t)$, numero de instancias del esquema H en la generacion t

$$P(h \in H) = \frac{m(H,t)f(H,t)}{\bar{Mf}(t)}$$

Teorema de Esquemas de Holland (Schema Theorem)

■ Lema 1

- Bajo selección proporcional al fitness, el número esperado de instancias del esquema H en el tiempo t es

$$E[m(H, t + 1)] = M \times P(h \in H) = \frac{m(H, t) f(H, t)}{\bar{f}(t)}$$

- Implicación:
 - Esquemas con fitness grande (bajo) que el fitness de la población promedio, tendrán mas (menos) probabilidad de aparecen en la siguiente generación
-

Teorema de Esquemas de Holland (Schema Theorem)

■ Operadores de búsqueda – un punto de cruce

- Modifica la distribución de los esquemas

| | | | | | | | | |
|---------|---|---|---|--|---|---|---|---|
| $h =$ | 1 | 0 | 1 | | 1 | 1 | 0 | 0 |
| $H_1 =$ | * | 0 | 1 | | * | * | * | 0 |
| $H_2 =$ | * | 0 | 1 | | * | * | * | * |

- Implicacion

- Esquemas con longitud grande son mas probables de ser cambiados con cruzamiento de un punto
-

Teorema de Esquemas de Holland (Schema Theorem)

■ Lema 2

- La probabilidad de que un esquema sobreviva a la siguiente generación es

$$P(\text{H sobrevive}) = 1 - P(\text{H no sobrevive})$$

$$1 - \frac{\delta(H)}{l-1} P_{diff}(H, t)$$

$P_{diff}(H, T)$, es la probabilidad de que el segundo padre no empareje con con el esquema H

Teorema de Esquemas de Holland (Schema Theorem)

■ Operadores de Búsqueda – Mutación (Inteligente)

- De tal manera de garantizar la supervivencia de un esquema, los genes diferentes de * deben permanecer sin cambios
- Probabilidad de no cambiar un gen
 - $(1-P_m)$
- Requiere que todos los $o(H)$ diferentes de * sobrevivan
 - $(1-P_m)^{o(H)}$
- Típicamente la probabilidad de aplicar el operador de mutación es aproximado mediante

$$(1 - p_m)^{o(H)} \approx 1 - o(H) p_m$$

Teorema de Esquemas de Holland (Schema Theorem)

■ Lema 3

- Bajo mutación de genes, la probabilidad (cota inferior) de un esquema H de orden $o(H)$ sobreviva en la generación t es:

$$1 - o(H) p_m$$

Teorema de Esquemas de Holland (Schema Theorem)

■ Teorema 1 Teorema de Esquemas

- El número esperado de esquemas H en la generación t+1, cuando usamos algoritmos genéticos canónicos, con selección proporcional, cruzamiento de un punto y mutación inteligente de genes es

$$E[m(H, t + 1)] \geq \frac{m(H, t) f(H, t)}{\bar{f}(t)} \left\{ 1 - p_c \frac{\delta(H)}{l} p_{diff}(H, t) - o(H) p_m \right\}$$

Teorema de Esquemas de Holland (Schema Theorem)

- El teorema de esquemas no provee una exacta vista del comportamiento de los algoritmos genéticos, y no puede predecir como un hiperplano específico es procesado en el tiempo
-

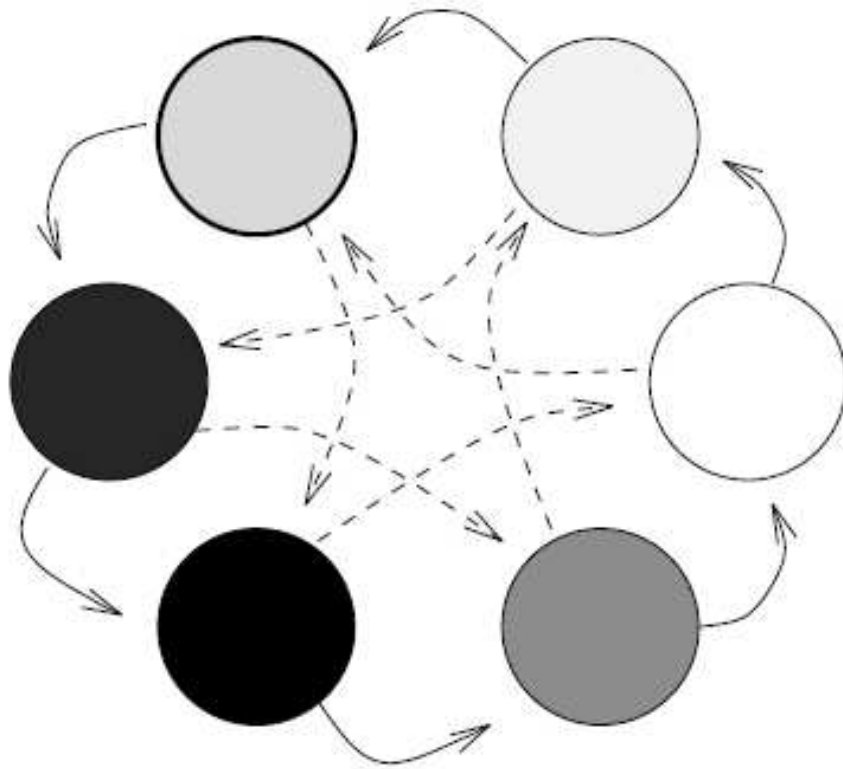
Otros Modelos de Computación Evolutiva

- **Evolution Strategies** y **Evolutionary Programming** son dos estrategias computacionales que usan búsquedas basadas en población
 - **Evolutionary Programming** basado en Artificial Intelligence Through Simulated Evolution 1966 L Fogel Owens and Walsh
-

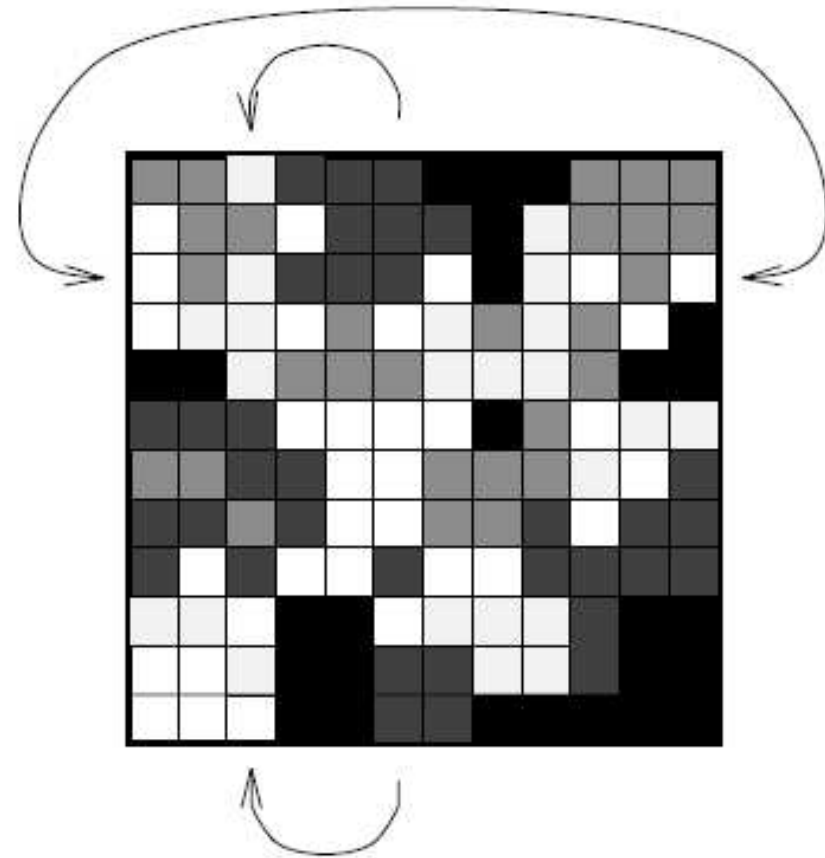
Otros Modelos de Computación Evolutiva

- Algoritmos estilo Genitor (steady state genetic algorithms)
 - CHC Cross generational elitist selection Heterogeneous recombination by incest prevention and Cataclysmic mutation
 - Algoritmos híbridos
 - Algoritmos genéticos paralelos
 - Island Models
 - Cellular Genetic Algorithms
-

Otros Modelos de Computación Evolutiva



An Island Model Genetic Algorithm



A Cellular Genetic Algorithm

Referencias

A Genetic Algorithm Tutorial

Darrell Whitley

Computer Science Department, Colorado State University
Fort Collins, CO 80523 whitley@cs.colostate.edu

Abstract

This tutorial covers the canonical genetic algorithm as well as more experimental forms of genetic algorithms, including parallel island models and parallel cellular genetic algorithms. The tutorial also illustrates genetic search by hyperplane sampling. The theoretical foundations of genetic algorithms are reviewed, include the schema theorem as well as recently developed exact models of the canonical genetic algorithm.

Keywords: Genetic Algorithms, Search, Parallel Algorithms
