

2

---

# ALGORITMOS GENÉTICOS

# ¿QUÉ SON LOS ALGORITMOS GENÉTICOS?

- ▶ Los **algoritmos genéticos** (GA) son una clase de algoritmos de optimización inspirados en el proceso de evolución natural.
- ▶ La técnica se basa en conceptos de “**Selección natural**” y “**Herencia genética**” (Darwin, 1859).
- ▶ Un **algoritmo genético** crea una **población** de **individuos** mediante la reproducción de los **progenitores**.

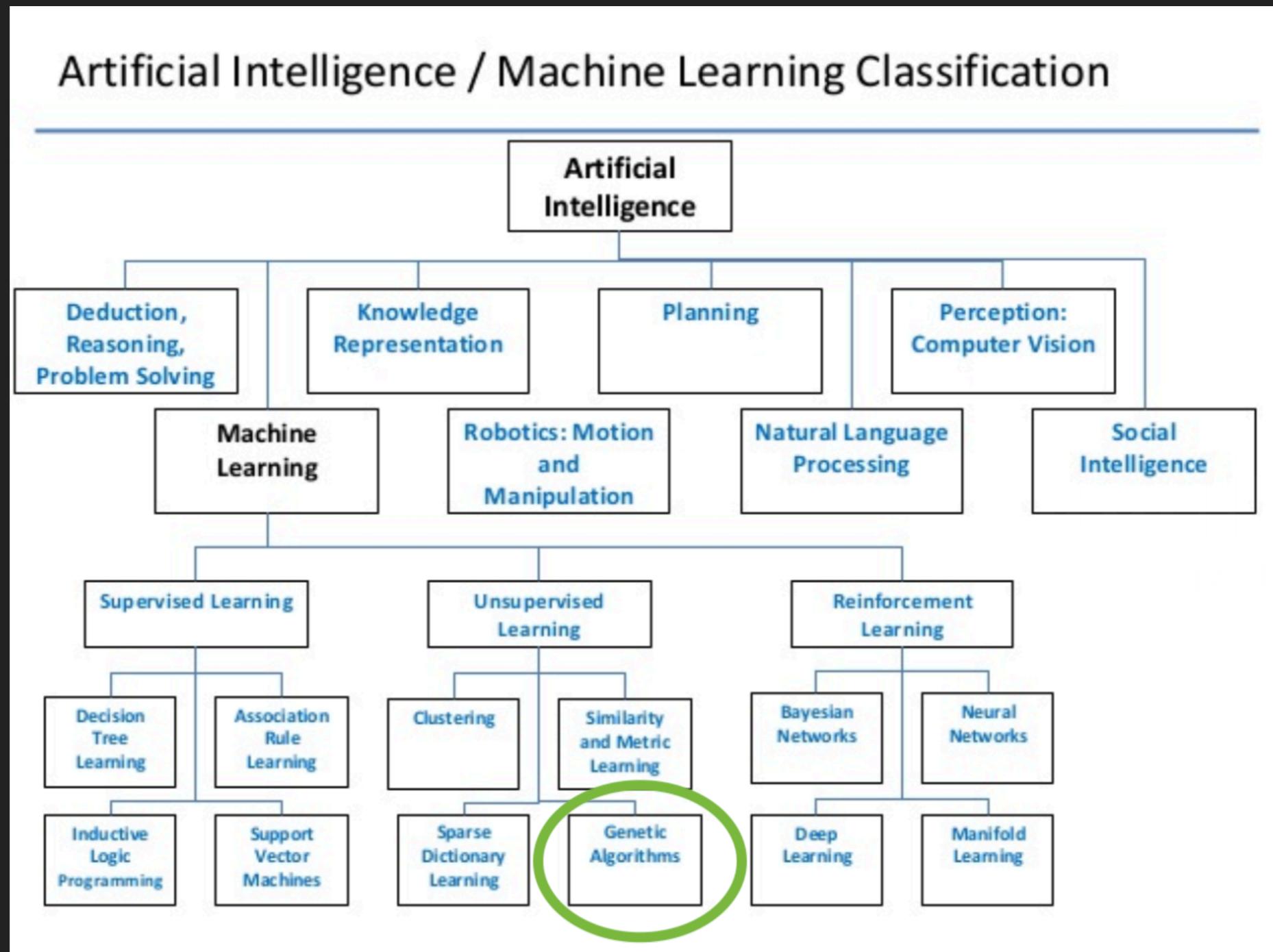
# ¿PARA QUÉ SIRVEN LOS ALGORITMOS GENÉTICOS?

- ▶ Se utilizan para encontrar **soluciones aproximadas** a problemas de **búsqueda** y **optimización**.

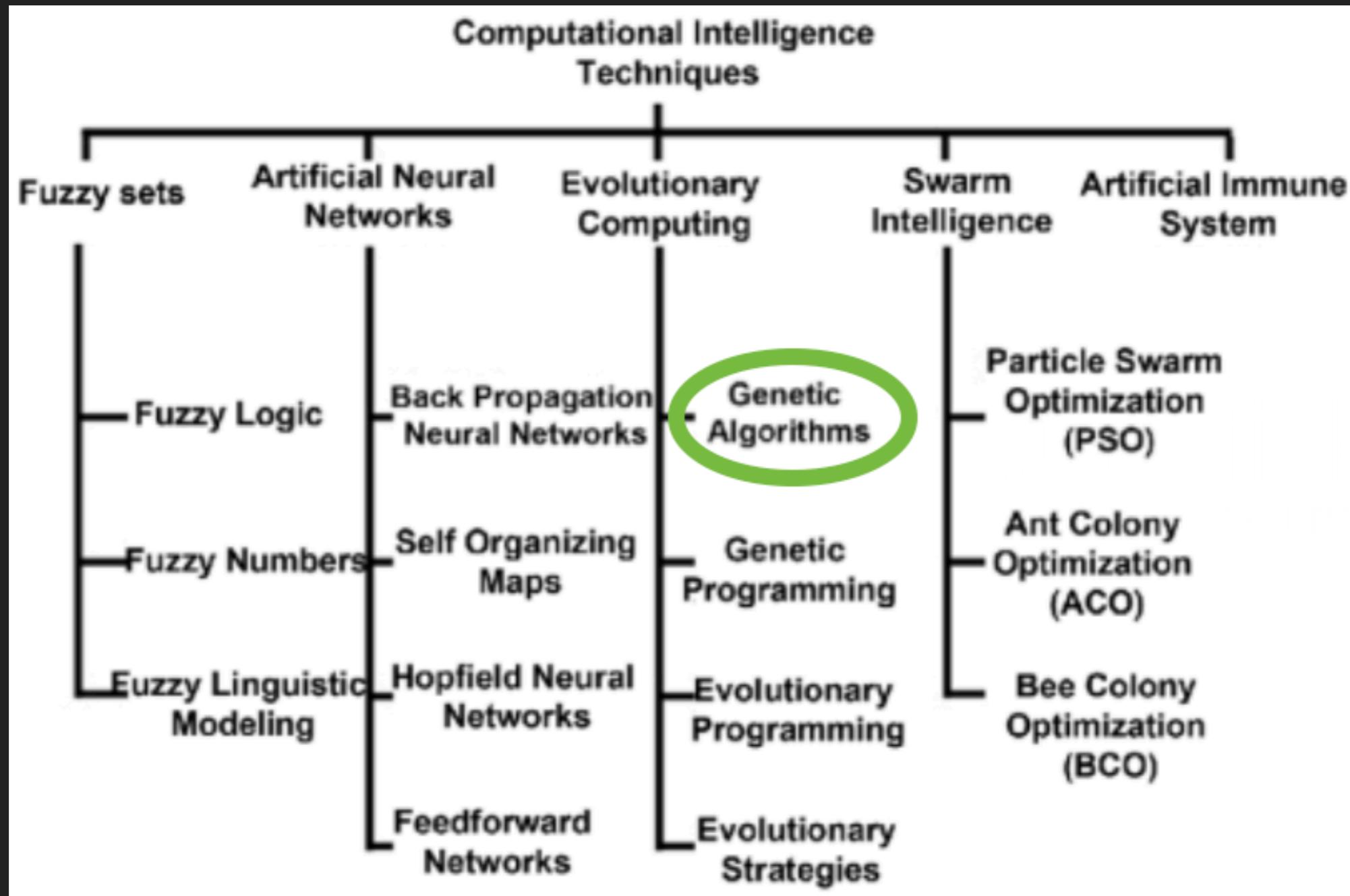
# ¿CUÁNDO Y QUIÉN DESARROLLÓ LOS ALGORITMOS GENÉTICOS?

- ▶ 1975 (John Henry Holland, 1929 - 2015)
- ▶ "Adaptation in natural and artificial systems."

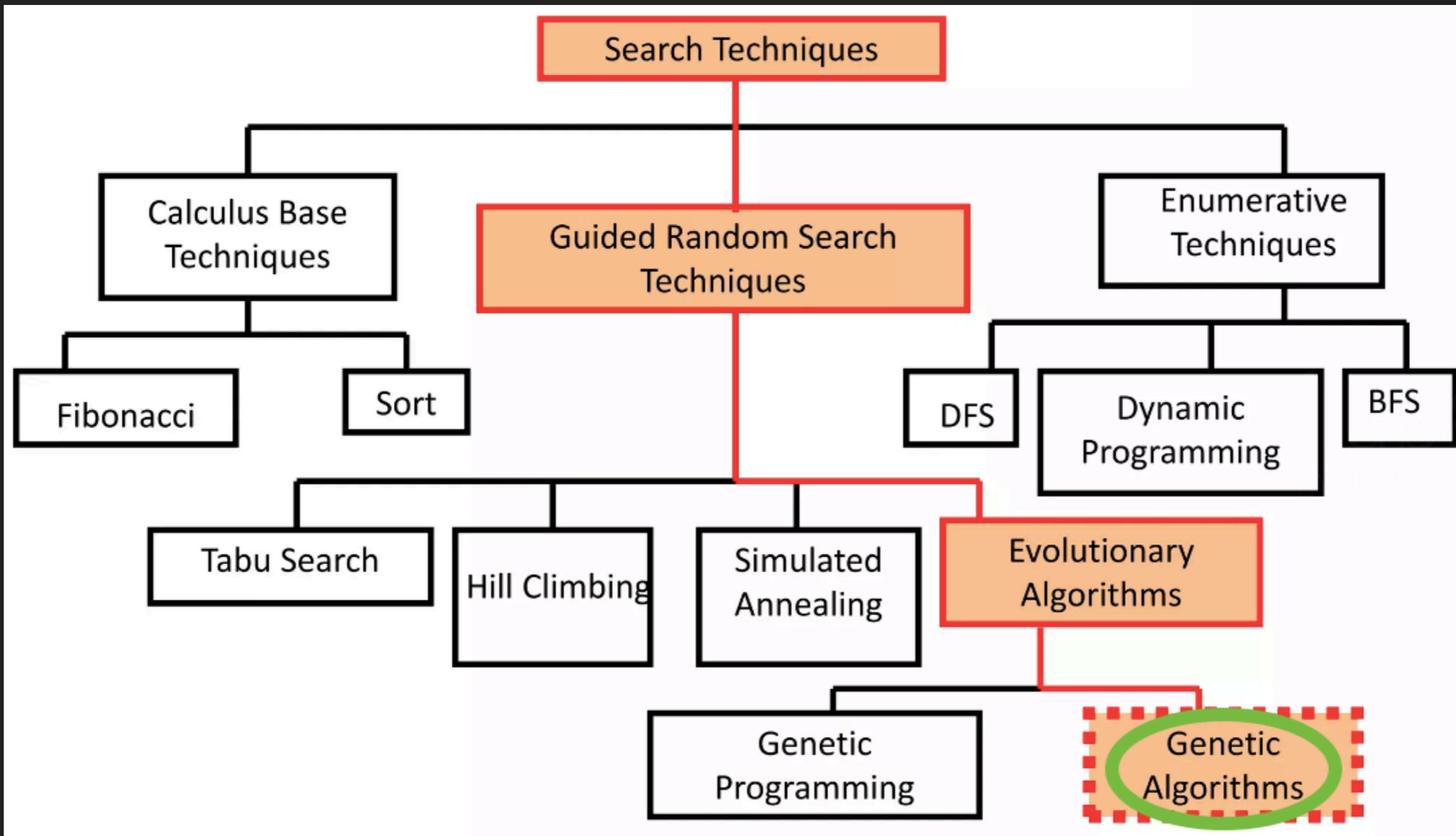
# ESQUEMA DE CLASIFICACIÓN EN EL CAMPO DE LA IA (I)



## ESQUEMA DE CLASIFICACIÓN EN EL CAMPO DE LA IA (II)



# ESQUEMA DE CLASIFICACIÓN EN EL CAMPO DE LA IA (III)



INGENIERÍA

# ESQUEMA DE CLASIFICACIÓN EN EL CAMPO DE LA IA (IV)

6	Popular Evolutionary Algorithm Variants .....	99
6.1	Genetic Algorithms .....	99
6.2	Evolution Strategies .....	101
6.3	Evolutionary Programming .....	103
6.4	Genetic Programming .....	104
6.5	Learning Classifier Systems .....	107
6.6	Differential Evolution .....	110
6.7	Particle Swarm Optimisation .....	112
6.8	Estimation of Distribution Algorithms .....	113

Eiben, A. E., & Smith, J. E. (2015). Introduction to evolutionary computing. Springer-Verlag Berlin Heidelberg.

# ¿EN QUÉ CONSISTE UN ALGORITMO GENÉTICO? - INICIALIZACIÓN

## INICIO

*INICIALIZAR* una población de individuos aleatoriamente  
*EVALUAR* cada individuo

**REPETIR HASTA QUE** (*CONDICIÓN DE TERMINACIÓN* es satisfecha)

- 1 *SELECCIONAR* progenitores
- 2 *RECOMBINAR* pares de progenitores
- 3 *MUTAR* los descendientes resultantes
- 4 *EVALUAR* los descendientes mutados

**FINREPETIR**

**FIN**



# ¿EN QUÉ CONSISTE UN ALGORITMO GENÉTICO? - INICIALIZACIÓN

**INICIO**

*INICIALIZAR una población de individuos aleatoriamente*

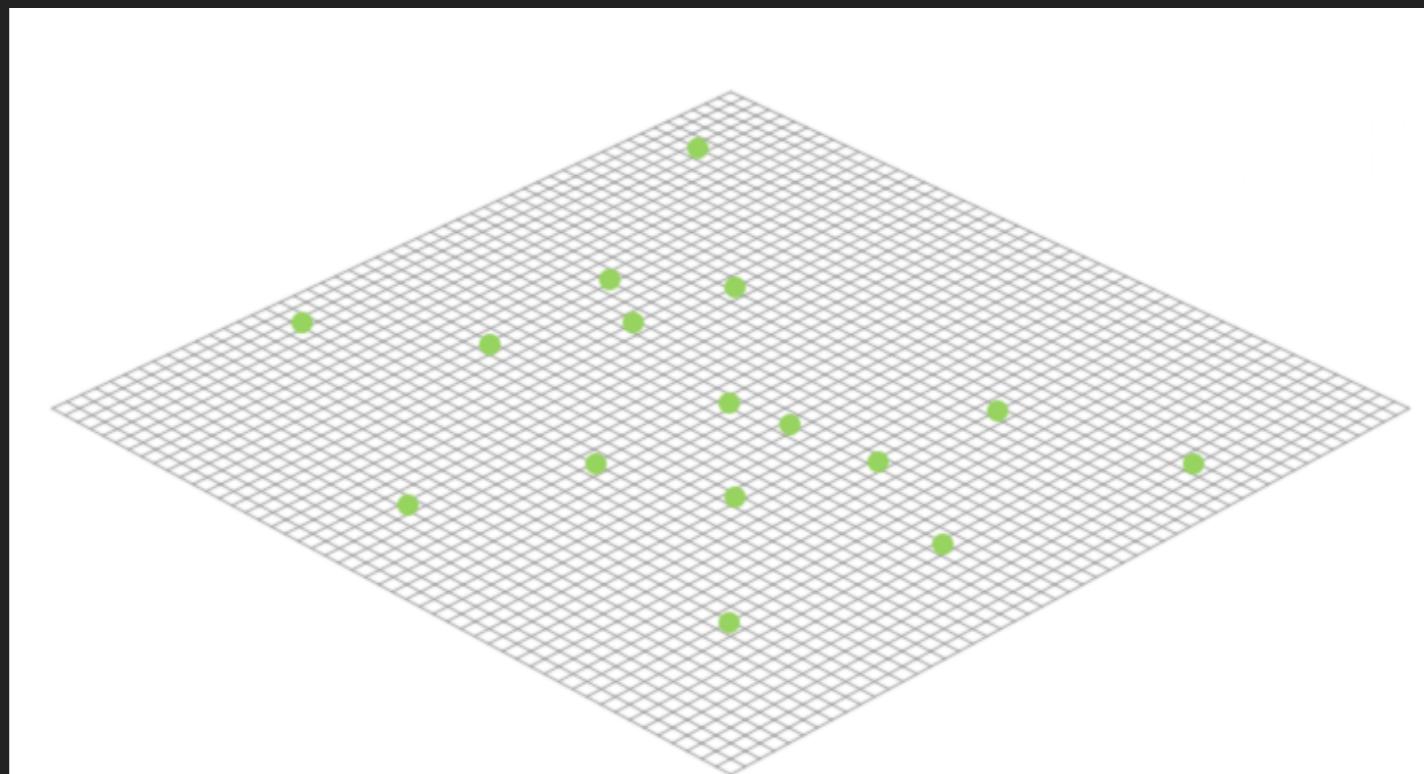
*EVALUAR cada individuo*

**REPETIR HASTA QUE** ( *CONDICIÓN DE TERMINACIÓN* es satisfecha)

- 1 *SELECCIONAR* progenitores
- 2 *RECOMBINAR* pares de progenitores
- 3 *MUTAR* los descendientes resultantes
- 4 *EVALUAR* los descendientes mutados

**FINREPETIR**

**FIN**



# ¿EN QUÉ CONSISTE UN ALGORITMO GENÉTICO? - EVALUACIÓN

**INICIO**

*INICIALIZAR* una población de individuos aleatoriamente

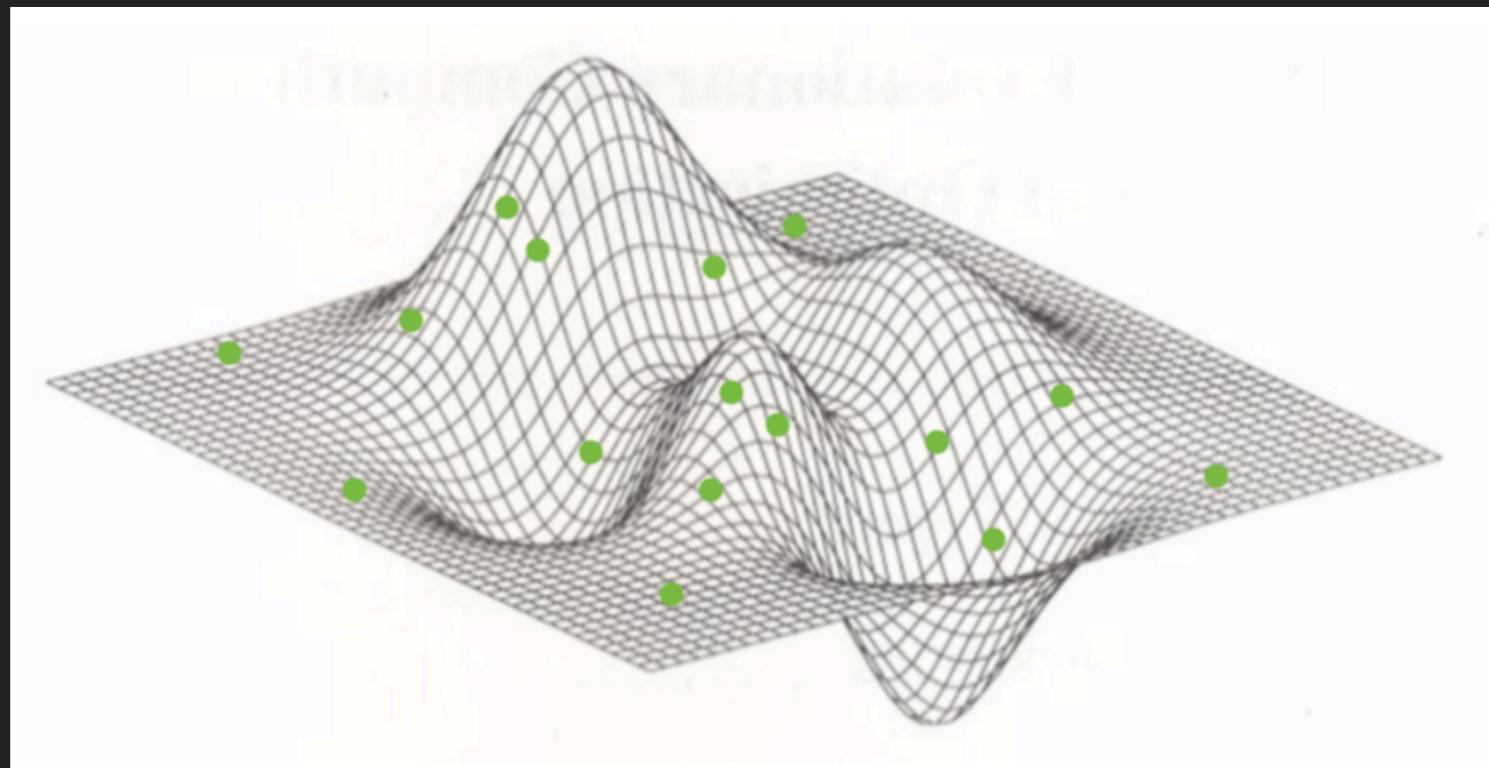
*EVALUAR* cada individuo

**REPETIR HASTA QUE** ( *CONDICIÓN DE TERMINACIÓN* es satisfecha)

- 1 *SELECCIONAR* progenitores
- 2 *RECOMBINAR* pares de progenitores
- 3 *MUTAR* los descendientes resultantes
- 4 *EVALUAR* los descendientes mutados

**FINREPETIR**

**FIN**



# ¿EN QUÉ CONSISTE UN ALGORITMO GENÉTICO? - EVALUACIÓN

**INICIO**

*INICIALIZAR* una población de individuos aleatoriamente  
*EVALUAR* cada individuo

**REPETIR HASTA QUE ( CONDICIÓN DE TERMINACIÓN es satisfecha)**

- 1 *SELECCIONAR* progenitores
- 2 *RECOMBINAR* pares de progenitores
- 3 *MUTAR* los descendientes resultantes
- 4 *EVALUAR* los descendientes mutados

**FINREPETIR**

**FIN**

# ¿EN QUÉ CONSISTE UN ALGORITMO GENÉTICO? - SELECCIÓN

**INICIO**

*INICIALIZAR* una población de individuos aleatoriamente

*EVALUAR* cada individuo

**REPETIR HASTA QUE** ( *CONDICIÓN DE TERMINACIÓN* es satisfecha)

    1 *SELECCIONAR* progenitores

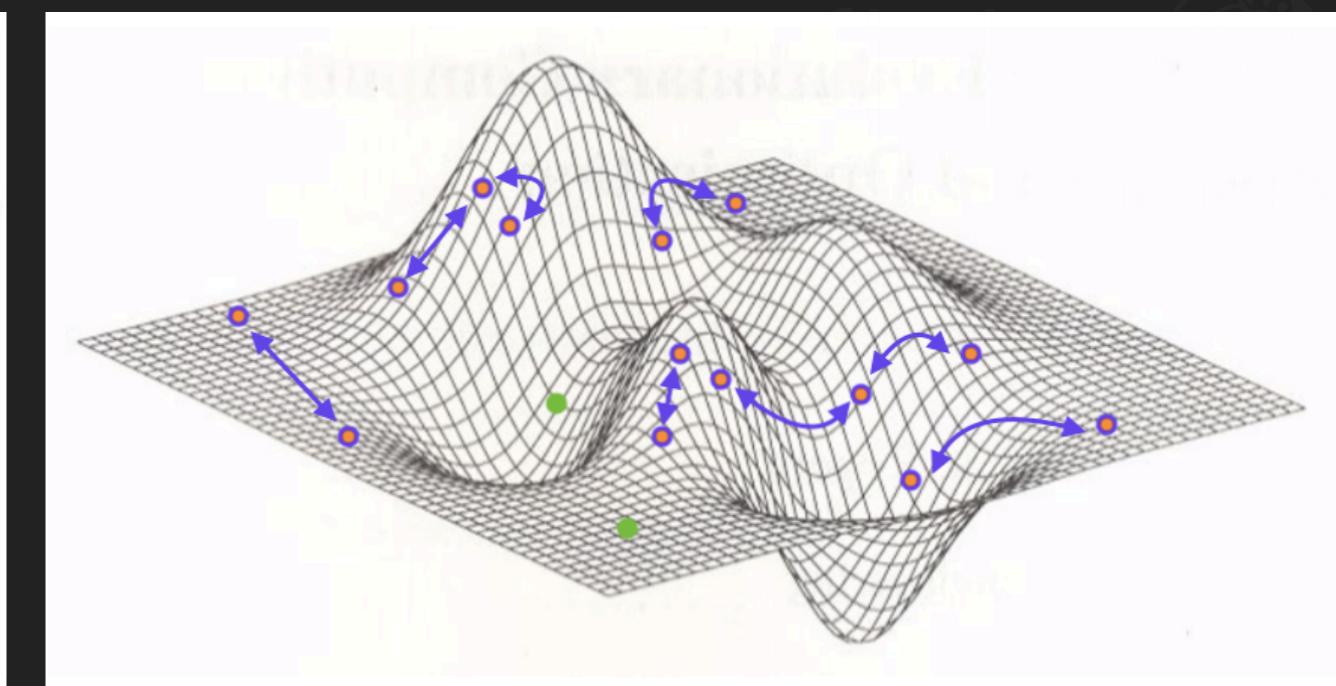
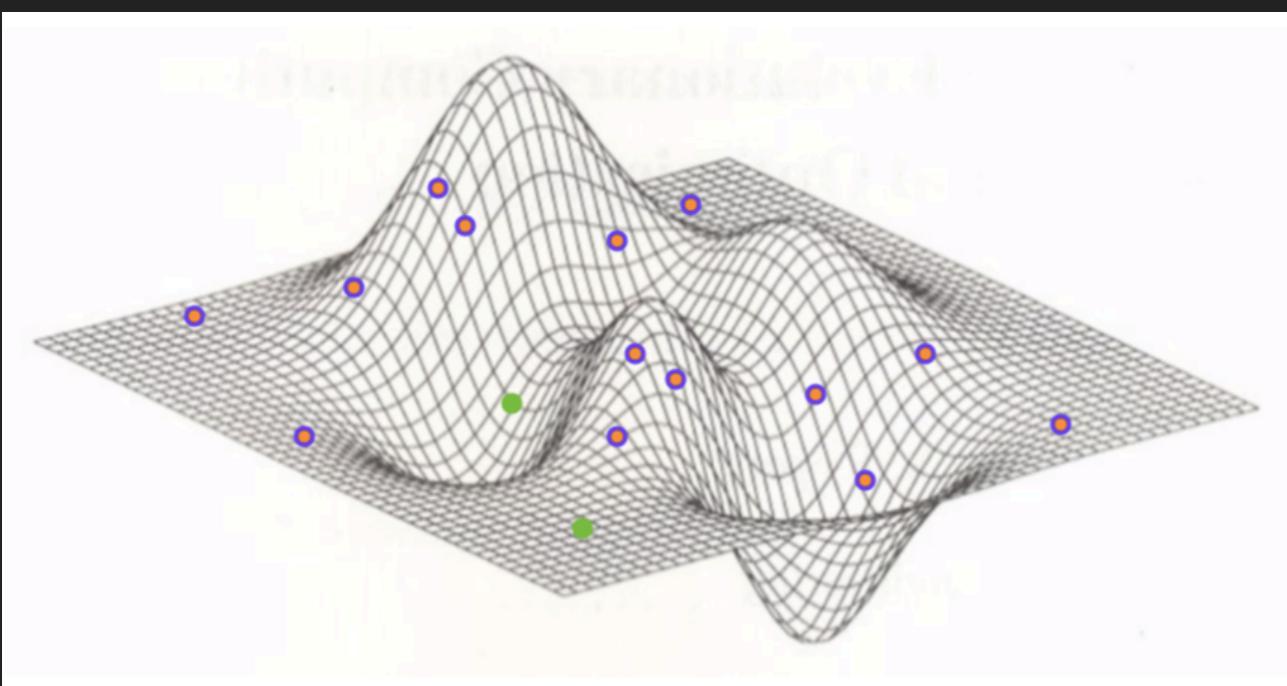
    2 *RECOMBINAR* pares de progenitores

    3 *MUTAR* los descendientes resultantes

    4 *EVALUAR* los descendientes mutados

**FINREPETIR**

**FIN**



# ¿EN QUÉ CONSISTE UN ALGORITMO GENÉTICO? - CRUZA

**INICIO**

*INICIALIZAR* una población de individuos aleatoriamente

*EVALUAR* cada individuo

**REPETIR HASTA QUE** ( *CONDICIÓN DE TERMINACIÓN* es satisfecha)

1 *SELECCIONAR* progenitores

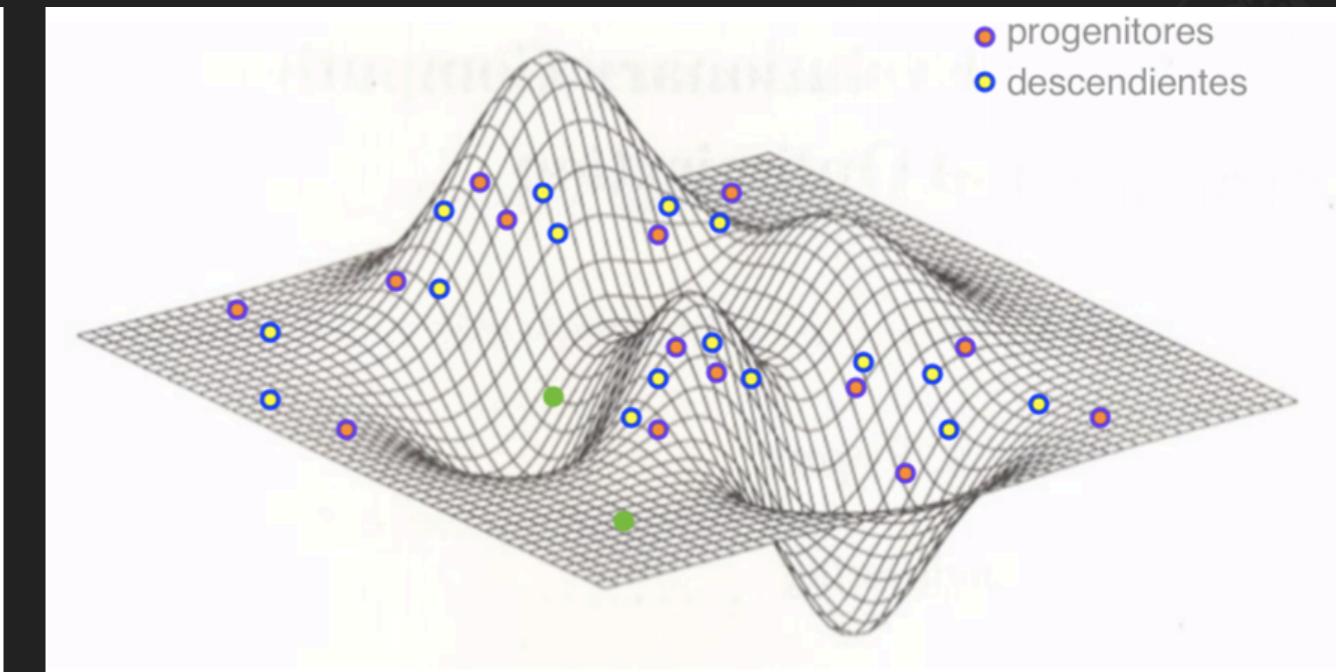
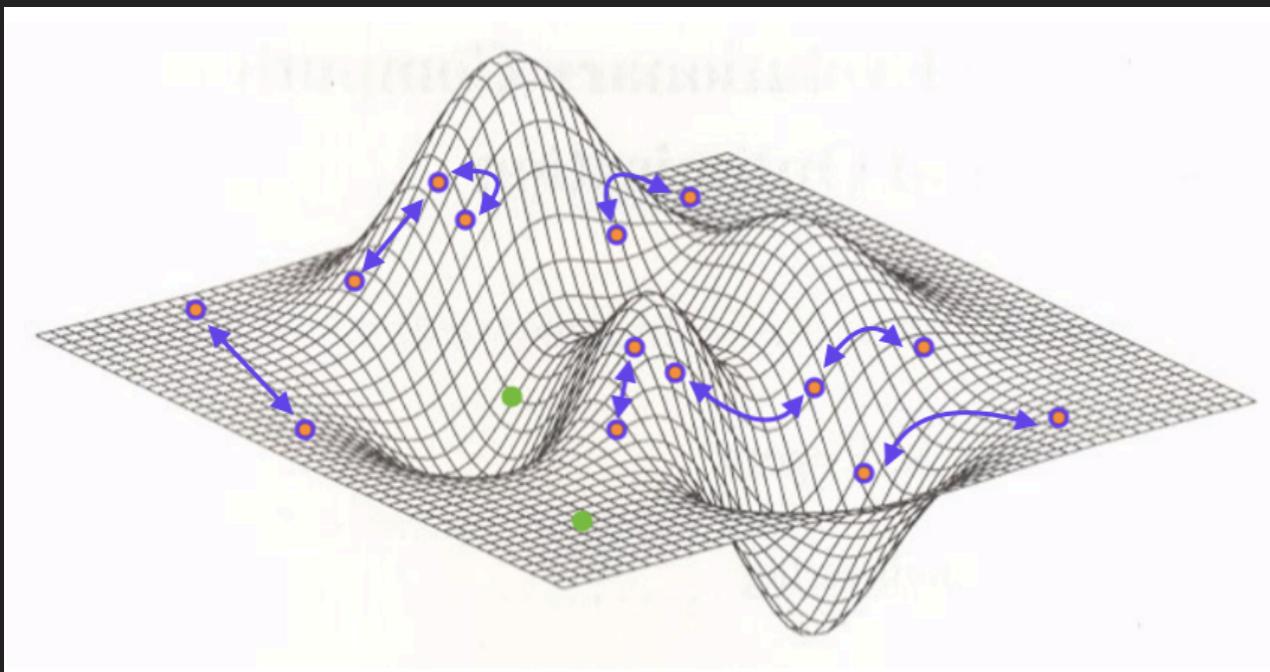
2 **RECOMBINAR** pares de progenitores

3 *MUTAR* los descendientes resultantes

4 *EVALUAR* los descendientes mutados

**FINREPETIR**

**FIN**



# ¿EN QUÉ CONSISTE UN ALGORITMO GENÉTICO? - MUTACIÓN

**INICIO**

*INICIALIZAR* una población de individuos aleatoriamente

*EVALUAR* cada individuo

**REPETIR HASTA QUE** ( *CONDICIÓN DE TERMINACIÓN* es satisfecha)

1 *SELECCIONAR* progenitores

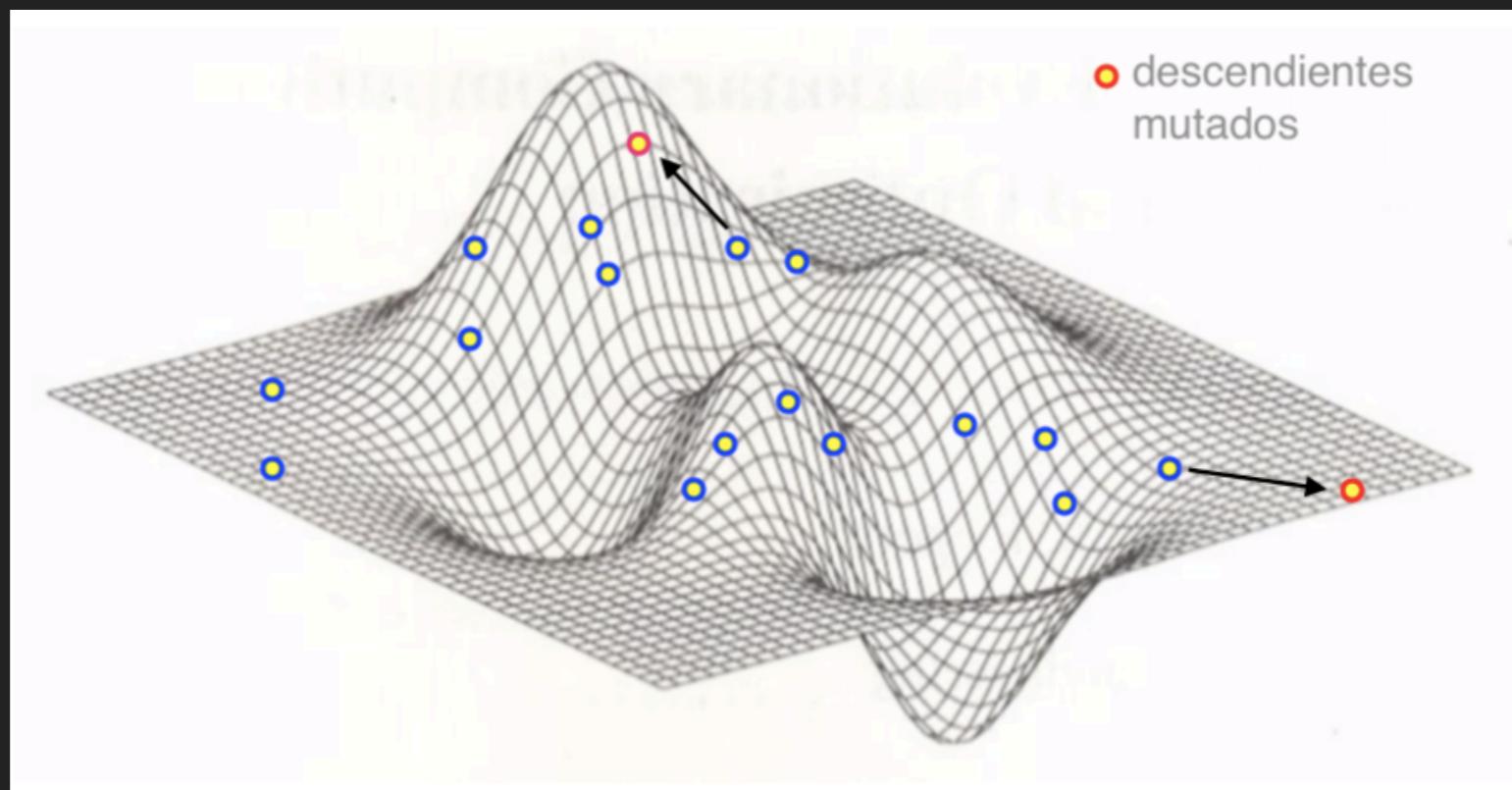
2 *RECOMBINAR* pares de progenitores

3 **MUTAR** los descendientes resultantes

4 *EVALUAR* los descendientes mutados

**FINREPETIR**

**FIN**



# ¿EN QUÉ CONSISTE UN ALGORITMO GENÉTICO? - EVALUACIÓN

**INICIO**

*INICIALIZAR* una población de individuos aleatoriamente

*EVALUAR* cada individuo

**REPETIR HASTA QUE** ( *CONDICIÓN DE TERMINACIÓN* es satisfecha)

1 *SELECCIONAR* progenitores

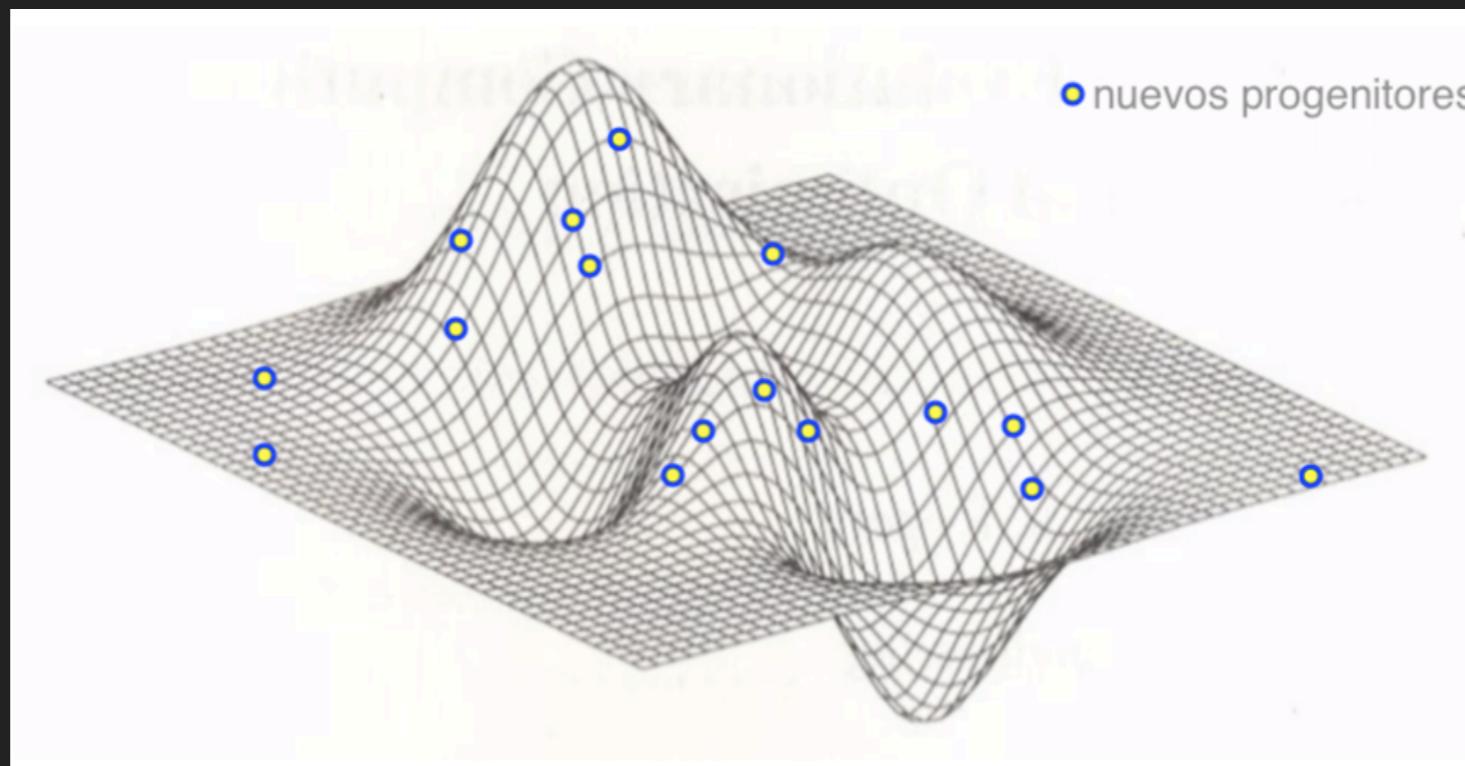
2 *RECOMBINAR* pares de progenitores

3 *MUTAR* los descendientes resultantes

4 *EVALUAR* los descendientes mutados

**FINREPETIR**

**FIN**

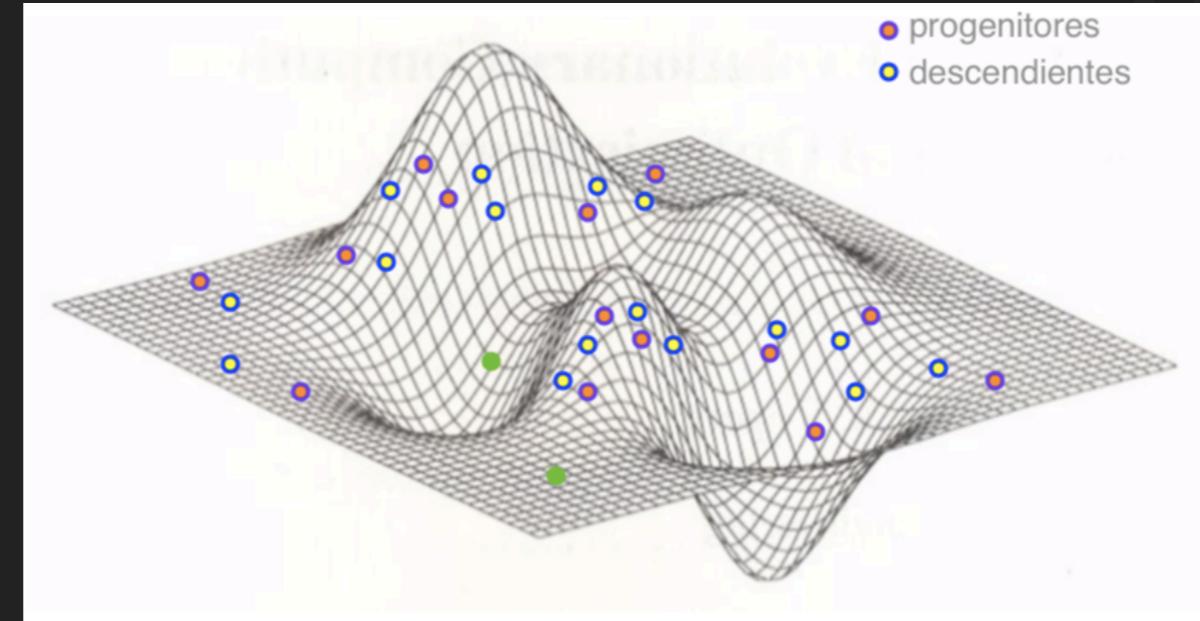


# ¿HASTA CUÁNDO DEBO ITERAR?

- ▶ Hasta encontrar el óptimo o un valor aproximado a él.
- ▶ ¿Como se que alcancé el óptimo?
- ▶ Criterios:
  - ✓ **Búsqueda:** Encontrar **cualquier** solución que cumpla con ciertos criterios.
  - ✓ **Optimización:** Encontrar **la mejor** solución posible según una función objetivo.

## MODELOS DE AG

- ▶ **Modelo generacional:** Durante cada iteración se crea una población completa con nuevos individuos. La nueva población reemplaza directamente a la antigua.
- ▶ **Modelo estacionario:** Durante cada iteración se escogen solo dos progenitores de la población y se les aplican los operadores genéticos. El/los descendiente/s reemplaza/n a uno/dos cromosoma/s de la población inicial



# CARACTERÍSTICAS

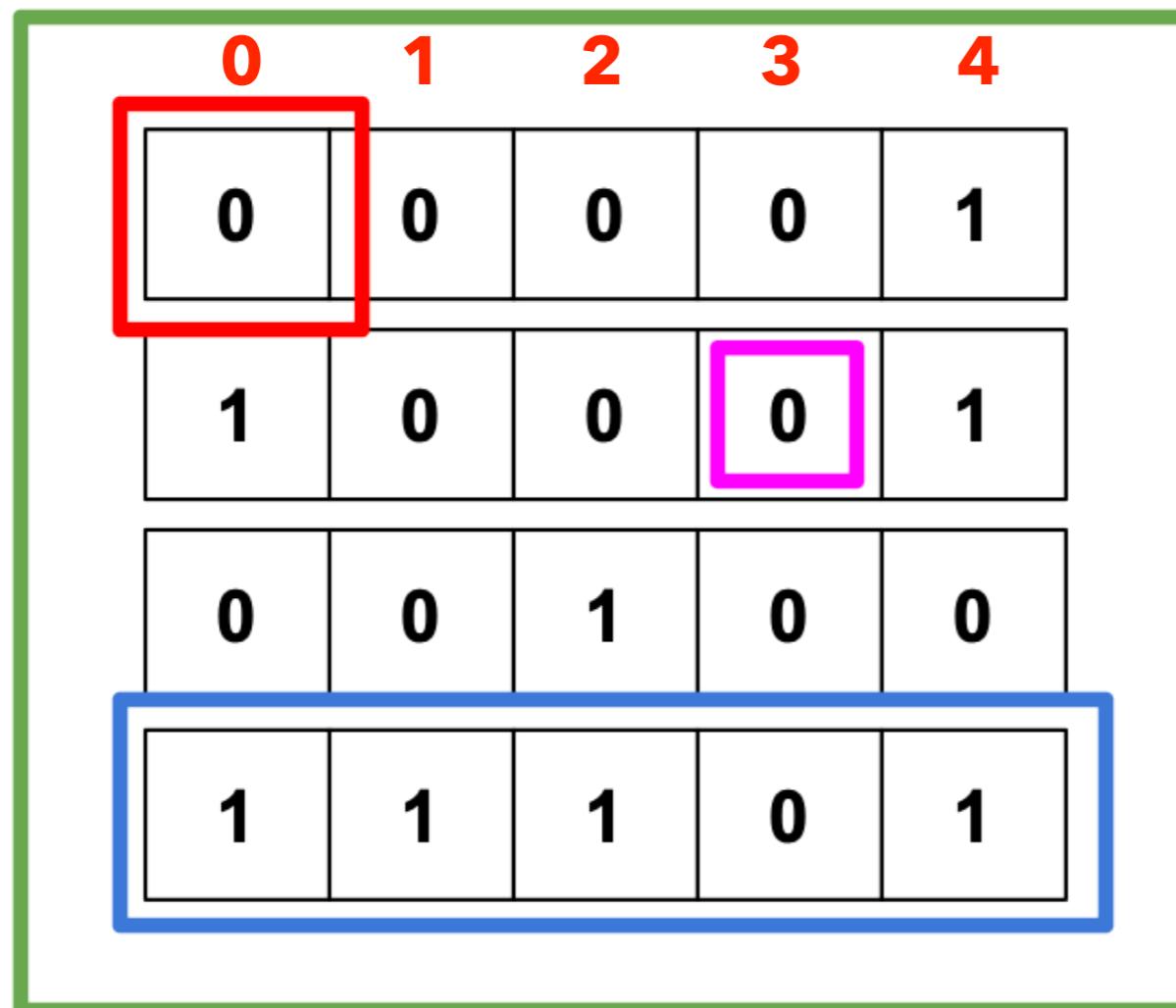
- ▶ Basados en **población**: los AG trabajan con una población de soluciones potenciales en lugar de una única solución.
- ▶ Orientados al **fitness**: las soluciones se evalúan en función de una función de fitness (**aptitud**) que cuantifica su calidad.
- ▶ **Iterativos**: el proceso se repite durante varias **generaciones** hasta que se cumple una condición de terminación.
- ▶ **Robustez**: los AG pueden manejar problemas de optimización continuos, discretos, combinatorios, multiobjetivo, etc.
- ▶ **Paralelismo**: el enfoque basado en la población permite el paralelismo en los cálculos.
- ▶ Buena **exploración y explotación**: los AG equilibran la exploración del espacio de búsqueda y la explotación de las mejores soluciones encontradas.

# COMO DESARROLLAR UN AG

- ▶ Elegir una **representación**
- ▶ Establecer una **población inicial** de individuos
- ▶ **Evaluar** los individuos
- ▶ Decidir cómo **seleccionar** los individuos para ser progenitores
- ▶ Aplicar operador de **cruce** adecuado
- ▶ Aplicar operador de **mutación** adecuado
- ▶ Definir cómo **reemplazar** a los progenitores
- ▶ Decidir la condición de **finalización** del algoritmo

## COMO REPRESENTAR INDIVIDUOS

# REPRESENTACIÓN DE INDIVIDUOS DE LA POBLACIÓN (I)



*Population*

*Chromosome*

*Gene*

*Allele*

## REPRESENTACIÓN DE INDIVIDUOS DE LA POBLACIÓN (II)

- ▶ **Población:** Es un conjunto de posibles soluciones (**cromosomas**).
- ▶ **Cromosoma:** Es una representación codificada de una posible solución al problema que se está resolviendo. Es una estructura que contiene **genes**. Se denomina también **individuo**.
- ▶ **Gen:** Es la unidad mínima de información en un **cromosoma**.
- ▶ **Alelo:** Representa un **valor** específico de un **gen** dentro de un cromosoma. Es uno de los posibles valores que un gen puede tomar.

## REPRESENTACIÓN DE INDIVIDUOS DE LA POBLACIÓN (III)

- ▶ Población:

[5, 2, 1, 8] <- Cromosoma 1

[2, 9, 0, 4] <- Cromosoma 2

[6, 7, 3, 8] <- Cromosoma 3

- ▶ Cromosoma 1: [5, 2, 1, 8]

- ▶ Gen: [Gen 1, Gen 2, Gen 3, Gen 4], es la posición en el vector del cromosoma.

- ▶ Alelo: El alelo del Gen1 es 5, el alelo del Gen 2 es 2, etc.

## REPRESENTACIÓN DE INDIVIDUOS DE LA POBLACIÓN (IV)

<b>Genotipo:</b>	<b>Fenotipo:</b>								
<table border="1"><tr><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td></tr></table>	1	0	1	0	0	0	1	1	= 163
1	0	1	0	0	0	1	1		

## REPRESENTACIÓN DE INDIVIDUOS DE LA POBLACIÓN (V)

- ▶ **Genotipo:** Representa la configuración interna de una solución candidata, generalmente codificada como una cadena de bits, números, o símbolos.
- ▶ **Fenotipo:** Es la manifestación externa y observable de un conjunto de genes (genotipo). Es la solución específica que resulta de los valores representados por el genotipo.

## REPRESENTACIÓN DE INDIVIDUOS DE LA POBLACIÓN (VI)

- ✓ Representación binaria
- ✓ Representación con números reales
- ✓ Representación de orden

## REPRESENTACIÓN DE INDIVIDUOS DE LA POBLACIÓN (VII)

- ▶ Representación binaria
- ▶ Los **individuos (cromosomas)** se representan como **cadenas binarias** de longitud fija.
- ▶ Cada **bit (alelo)** de la cadena representa una parte de la solución.
- ▶ Ejemplo:  $I_1 = 10110010$
- ▶  $I_1_{\text{entero}} = 1 \cdot 2^7 + 0 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 128 + 32 + 16 + 2 = 178$
- ▶ Los individuos binarios pueden representar números con n decimales de precisión.

## REPRESENTACIÓN DE INDIVIDUOS DE LA POBLACIÓN (VIII)

- ▶ Representación con números reales
- ▶ Los **individuos** se representan como matrices de números reales.
- ▶ Ejemplo:  $I2 = [3.5, 2.1, -1.4, 0.9]$

## REPRESENTACIÓN DE INDIVIDUOS DE LA POBLACIÓN (IX)

- ▶ Representación de orden
- ▶ Los **individuos** se representan con permutaciones.

$$I3 = [7, 2, 4, 9, 3, 1, 5]$$

- ▶ Se emplea en problemas de secuenciación.
- ▶ Ejemplo: Viajante de Comercio. Las ciudades se identifican con números naturales.

## FUNCTION FITNESS - FUNCTION OBJETIVO

## QUE ES LA FUNCIÓN FITNESS (APTITUD)

- ▶ La función de **aptitud** es aquella función específica de un problema que evalúa y asigna una puntuación de **aptitud** a cada individuo de la población.
- ▶ La puntuación de **aptitud** indica qué tan bien un individuo resuelve un problema.

## FUNCIÓN FITNESS Y FUNCIÓN OBJETIVO (II)

- ▶ Para un problema de **maximización**, la función de aptitud podría ser directamente el valor de la **función objetivo**.
- ▶ Para un problema de **minimización**, podría ser el **inverso** del valor de la función objetivo.

## FUNCIÓN FITNESS Y FUNCIÓN OBJETIVO (II)

- ▶ Ejemplo A: Supongamos que tenemos una función objetivo  $f(x) = x^2$
- ▶ Se busca **maximizar**  $f(x)$
- ▶ Si un cromosoma  $x$  (convertido a enteros) tiene un valor de  $x=5$ , =>  $f(5) = 25$
- ▶ La **aptitud** de este cromosoma será 25, es decir,  $\text{aptitud}(x)=f(x)$

## FUNCIÓN FITNESS Y FUNCIÓN OBJETIVO (III)

- ▶ Ejemplo B: Supongamos que tenemos una función objetivo  $f(x) = x^2$
- ▶ Se busca **minimizar**  $f(x)$
- ▶ Si un cromosoma  $x$  (convertido a enteros) tiene un valor de  $x=5$ ,  $\Rightarrow f(5) = 25$

Individuo A:

$x=1, f=1$  (buena solución)

Individuo B:

$x=5, f=25$  (mala solución)

- ▶ La **aptitud** de este cromosoma podría ser el inverso del valor de la función objetivo:  
 $\text{aptitud}(x) = 1/(f(x) + \epsilon), \epsilon > 0$
- ▶ La **aptitud** de este cromosoma será:  $1/(25 + 1) = 1/26 = 0.0385$

## POBLACIÓN

## POBLACIÓN INICIAL

- ▶ La **población inicial** es un conjunto de individuos generado aleatoriamente al ejecutar el AG.
- ▶ El **tamaño de la población** es el número de individuos que participarán del proceso de búsqueda de una solución.
- ▶ Ejemplo: P es una población formada por 5 individuos.

$$P = [10110010, 11001101, 00110101, 11100010, 00011110]$$

## OPERADORES GENÉTICOS

## OPERADORES GENÉTICOS

- ▶ Los **operadores genéticos** se utilizan para gestionar la evolución de la población.
- ▶ Los principales operadores genéticos son:
  - ✓ Selección
  - ✓ Cruza
  - ✓ Mutación

## OPERADORES GENÉTICOS - SELECCIÓN

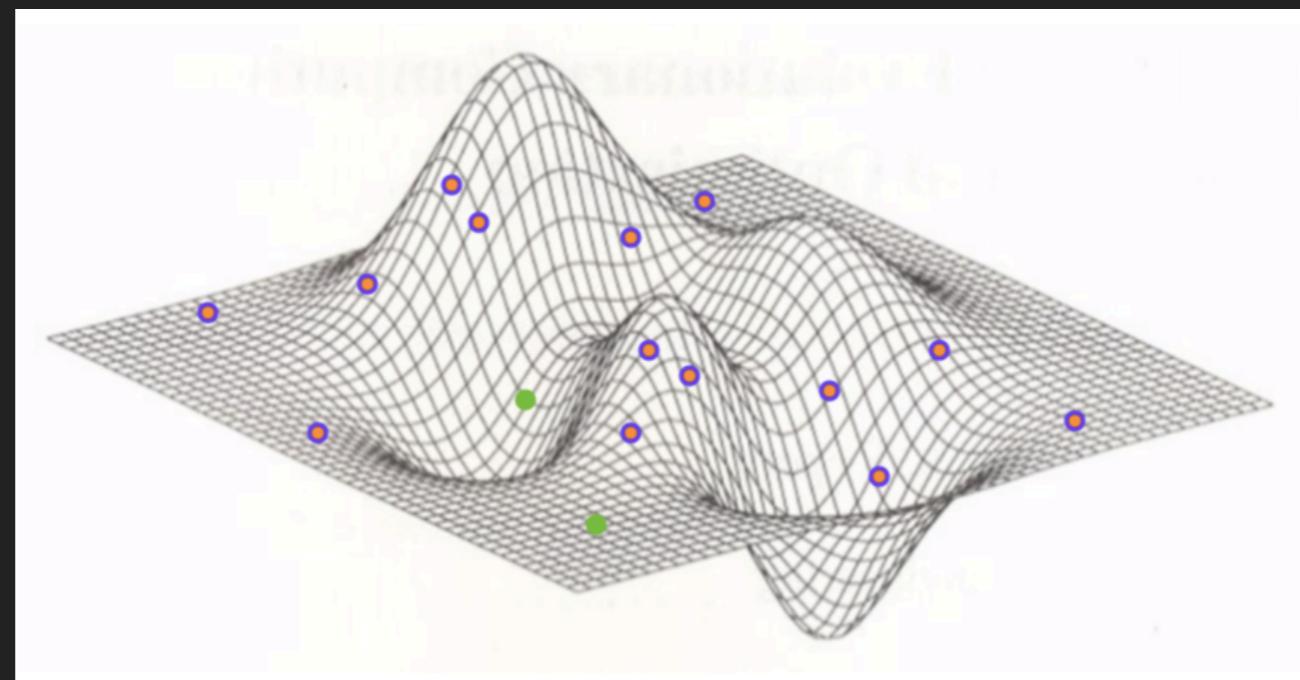
✓ Selección

✓ Cruza

✓ Mutación

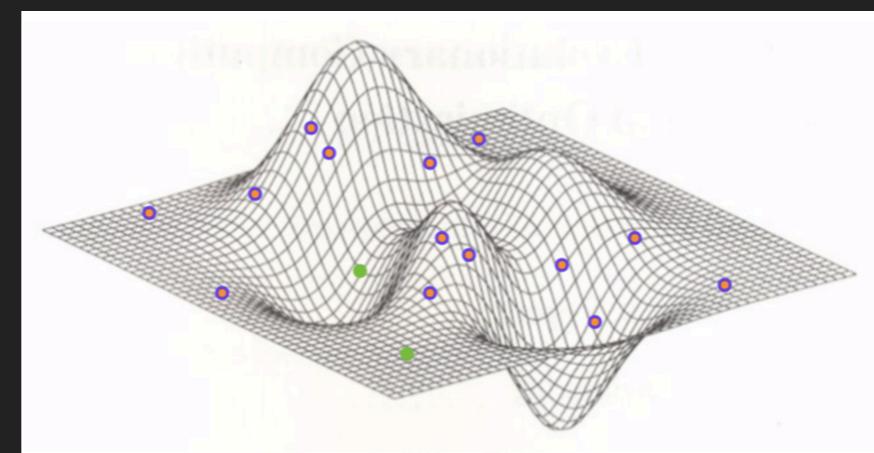
## OPERADORES GENÉTICOS - SELECCIÓN

- ▶ ¿Para qué sirve la Selección?
- ▶ Para que algunos individuos de la población se reproduzcan.



## OPERADORES GENÉTICOS - ¿QUÉ ES PRESIÓN SELECTIVA?

- ▶ Mide cuán "exigente" es el operador de selección para elegir a los mejores individuos.
- ▶ Alta presión selectiva:
  - ✓ Los individuos con fitness alto dominan rápidamente.
  - ✓ Riesgo: Convergencia prematura (poca diversidad).
- ▶ Baja presión selectiva:
  - ✓ Individuos menos aptos aún tienen probabilidad de ser seleccionados.
  - ✓ Riesgo: Evolución lenta.



## OPERADORES GENÉTICOS - SELECCIÓN

- ▶ Operadores de selección:

- ✓ Selección por Ruleta (Roulette wheel selection RWS  
o Fitness proportionate selection FPS)
- ✓ Selección por Torneo (Tournament selection TS)
- ✓ Selección por Ranking (Linear rank selection LRS)
- ✓ Otros

## OPERADORES GENÉTICOS - SELECCIÓN

- ✓ Selección por Ruleta
- ✓ Selección por Torneo
- ✓ Selección por Ranking

## OPERADOR DE SELECCIÓN - SELECCIÓN PROPORCIONAL DE APTITUD (FPS)

- ▶ También llamado “**Roulette Wheel**” .
- ▶ El FPS es el operador de selección original de Holland.
- ▶ La probabilidad de que un individuo sea seleccionado para reproducirse es la aptitud de ese individuo dividida por la sumatoria de aptitudes de la población.

$$P_s = \frac{f(i)}{\sum_{i=1}^N f(i)}$$

## OPERADOR DE SELECCIÓN - SELECCIÓN PROPORCIONAL DE APTITUD (FPS)

- ▶ Este método (**ruleta**) está basado en el conocimiento de toda la población.
- ▶ Cuando la **población es muy grande** o **está muy dispersa** la selección por **ruleta** requiere mucho tiempo de procesamiento (> costo computacional).

## OPERADOR DE SELECCIÓN - SELECCIÓN PROPORCIONAL DE APTITUD (FPS)

- ▶ Supongamos tres individuos con aptitudes

$$f = [1, 2, 3]$$

- ▶ Probabilidades:

$$P_{sel} = [1/6, 2/6, 3/6] = [0.167, 0.333, 0.5]$$

- ▶ Probabilidades acumuladas:

$$P_{acc} = [0.167, 0.5, 1.0]$$

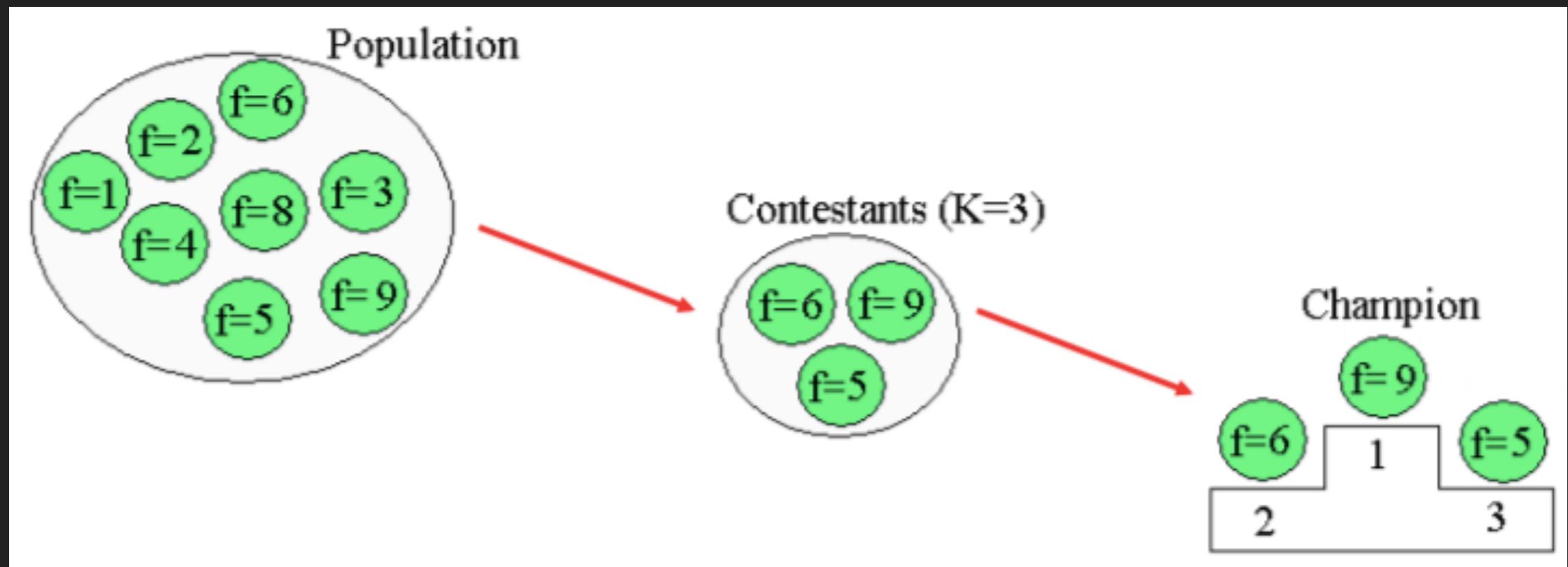
- ▶ Si genero un número random  $r = 0.4 \rightarrow$  cae en el intervalo del segundo individuo ( $0.167 < 0.4 \leq 0.5$ )  $\rightarrow$  se selecciona el individuo 2.



## OPERADORES GENÉTICOS - SELECCIÓN

- ✓ Selección por Ruleta
- ✓ Selección por Torneo
- ✓ Selección por Ranking

## OPERADOR DE SELECCIÓN - SELECCIÓN POR TORNEO (I)



## OPERADOR DE SELECCIÓN - SELECCIÓN POR TORNEO (II)

$I_1 = 100011101 <- f(I_1)$

$I_2 = 000010010$

$I_3 = 101011010$

$I_4 = 010111010 <- f(I_4)$

Comparo:  $f(I_1) > f(I_4)$

Se selecciona  $f(I_1)$  si el problema es de maximización

## OPERADOR DE SELECCIÓN - SELECCIÓN POR TORNEO (III)

### ► Selección por torneos

- ✓ No requiere ningún conocimiento de la población, ni una medida cuantificable de calidad.
- ✓ Se basa en comparar y clasificar a dos o más individuos cualesquiera.

## OPERADORES GENÉTICOS - SELECCIÓN

- ✓ Selección por Ruleta
- ✓ Selección por Torneo
- ✓ Selección por Ranking

## OPERADORES GENÉTICOS - SELECCIÓN

### ✓ Selección por Ranking

- Linear Ranking Selection
- Exponential Ranking Selection
- Boltzmann Selection
- Otros

## OPERADORES GENÉTICOS - SELECCIÓN

- Linear Ranking Selection

Individual	Fitness	Rank	$P_{selLR} \ (s = 2)$	$P_{selLR} \ (s = 1.5)$
A	1	0	0	0.167
B	4	1	0.33	0.33
C	5	2	0.67	0.5

- Se genera un número aleatorio  $r$ , si  $r \leq P_{selLR}$  se elige al individuo de rango  $i$ .

# OPERADOR DE SELECCIÓN - SELECCIÓN POR RANKING (I)

## ► Selección por ranking lineal (pasos)

- ✓ Se ordena la población por aptitud en forma ascendente.
- ✓ Se asigna Rango = 0 al peor individuo y Rango = N-1 al mejor.
- ✓ Se calcula la probabilidad de selección del individuo  $i$  según:

$$P_i = \frac{2 - s}{N} + \frac{2i(s - 1)}{N(N - 1)}$$

donde:

- $P_i$  es la probabilidad de selección del individuo de rango  $i$ ,
- $s$  es el parámetro de selección ( $1 < s \leq 2$ ),
- $N$  es el tamaño de la población,
- $i$  es la posición o rango del individuo (0 es el peor y  $N-1$  es el mejor)

## OPERADOR DE SELECCIÓN - OTROS

- ▶ Ademas de Roulette, Torneo y Ranking existen otros operadores de selección.

## OPERADOR DE SELECCIÓN - OTROS

### ► 1. Selección por Umbral (Threshold Selection)

- ✓ Solo se seleccionan individuos cuyo fitness supera un umbral predefinido.
- ✓ Útil cuando se busca mantener un nivel mínimo de calidad en la población.

### ► 2. Selección por Ventana (Window Selection)

- ✓ Ajusta los valores de fitness restando el peor valor de la población antes de aplicar otro operador (como la ruleta o ranking).
- ✓ Ayuda a evitar problemas cuando hay valores de fitness negativos o muy dispersos.

### ► 3. Selección por Truncamiento (Truncation Selection)

- ✓ Se eligen solo los mejores individuos (por ejemplo, el top 10% de la población).
- ✓ Muy utilizada en estrategias evolutivas (Evolution Strategies, ES).

## OPERADOR DE SELECCIÓN - OTROS

### ► 4. Selección por Competición (Tournament Selection variantes)

- ✓ Torneo **determinístico**: Siempre se elige el mejor de un subconjunto aleatorio.
- ✓ Torneo **probabilístico**: El ganador se elige con una probabilidad basada en su fitness (no siempre gana el mejor).
- ✓ Torneo de **tamaño variable**: El tamaño del torneo puede variar para ajustar la presión selectiva.

### ► 5. Selección por Exclusión (Exclusion Selection)

- ✓ Evita seleccionar individuos demasiado similares (promueve la diversidad).
- ✓ Se puede combinar con otros operadores para mantener variedad en la población.

### ► 6. Selección por Estado Estacionario (Steady-State Selection)

- ✓ En lugar de reemplazar toda la población en cada generación, solo se reemplazan unos pocos individuos (los peores).
- ✓ Los nuevos individuos compiten directamente con los existentes.

## OPERADOR DE SELECCIÓN - OTROS

### ► 7. Selección por Muestreo Determinístico (Deterministic Sampling)

- ✓ Se asignan copias de individuos según su fitness (ej: un individuo con fitness 3 tiene 3 copias si el fitness promedio es 1).
- ✓  $\text{Num copias} = \text{Fitness}_i / \text{Fitness\_promedio}$
- ✓ Luego se selecciona aleatoriamente de este conjunto ampliado (pool).

### ► 8. Selección por Hipervolumen (Hypervolume-Based Selection)

- ✓ Usado en algoritmos multiobjetivo (como NSGA-II). Selecciona soluciones que maximizan el hipervolumen en el espacio de objetivos.
- ✓ El hipervolumen es una métrica utilizada en optimización multiobjetivo para evaluar la calidad y diversidad de un conjunto de soluciones.

### ► 9. Selección por Distancia Crowding (Crowding Distance)

- ✓ Prioriza individuos únicos o poco representados en el espacio de búsqueda para mantener diversidad.

## OPERADOR DE SELECCIÓN - OTROS

### ► 10. Selección por Boltzmann

- ✓ Inspirada en la termodinámica: la probabilidad de selección depende de una "temperatura" que disminuye con el tiempo, permitiendo mayor exploración al inicio y explotación después.
- ✓ + temperatura => + aleatoriedad (es mas "creativo") => + exploración
- ✓ - temperatura => favorece a mejores individuos (es mas determinista) => + explotación

### ► 11. Selección por Jerarquía (Hierarchical Selection)

- ✓ Agrupa individuos en niveles (ej: basado en clusters de fitness) y selecciona proporcionalmente de cada grupo.

## OPERADORES GENÉTICOS - CRUZA

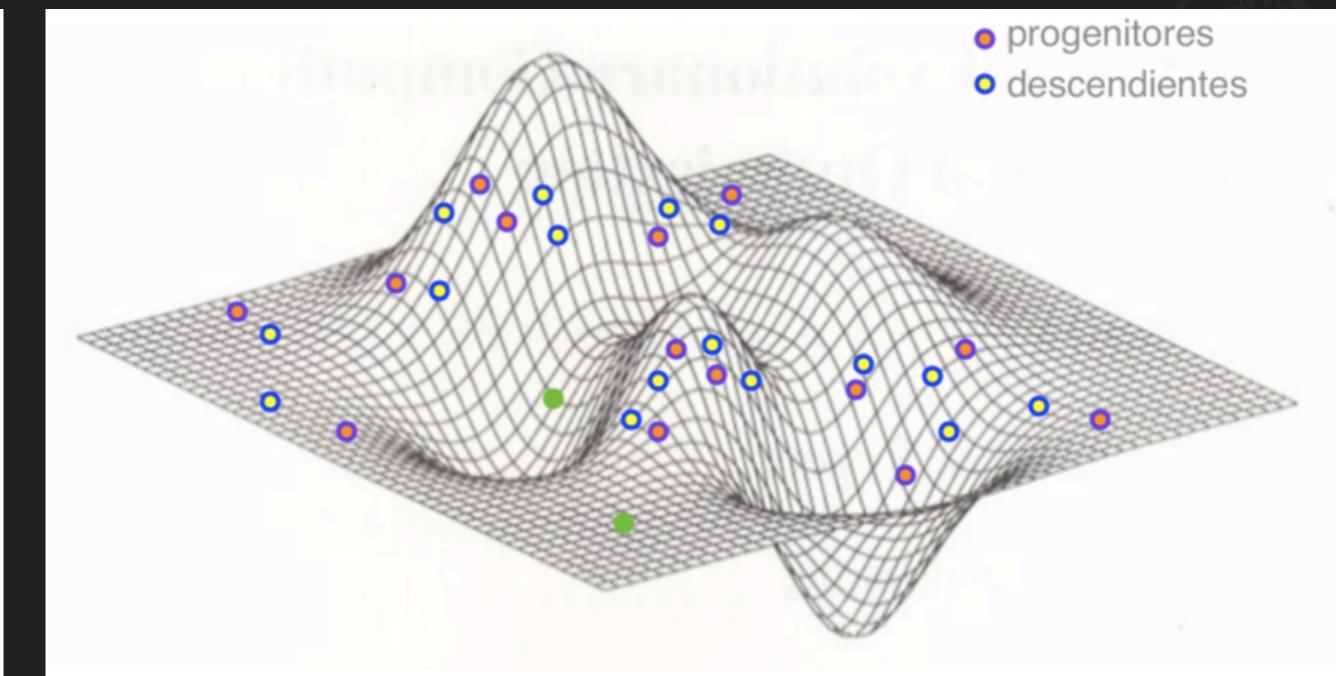
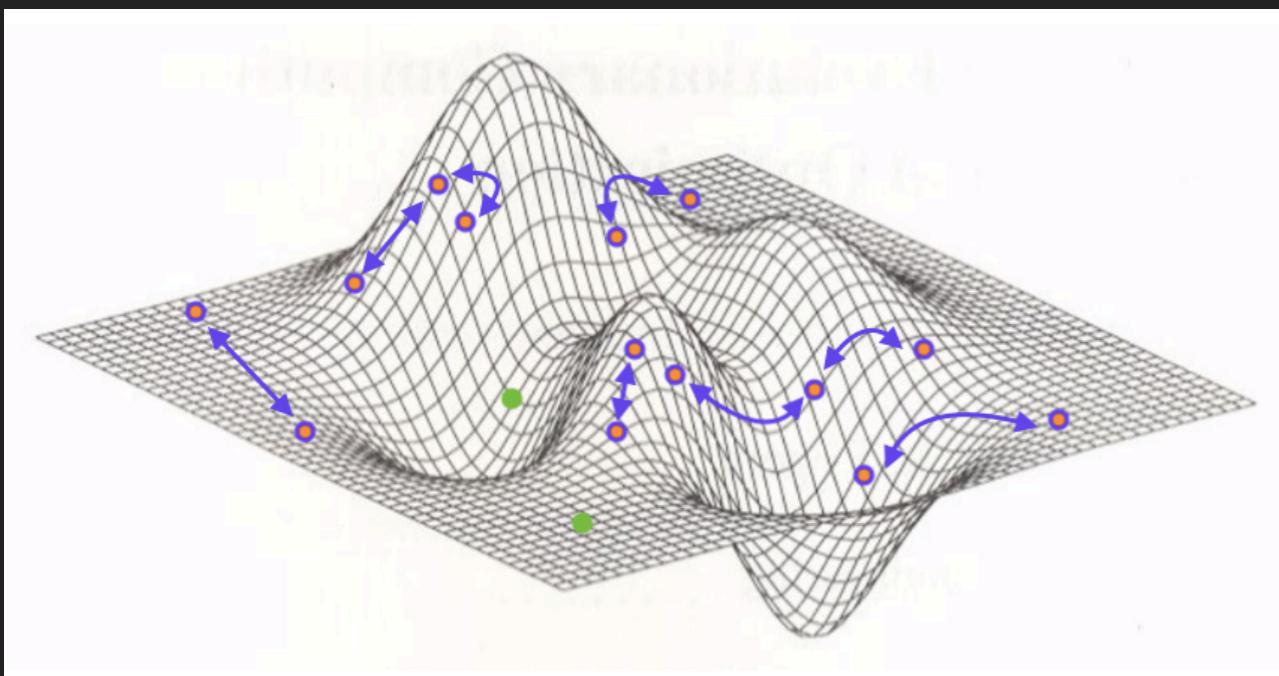
✓ Selección

✓ Cruza

✓ Mutación

## OPERADORES GENÉTICOS - CRUZA

- ▶ ¿Para qué sirve la **Cruza**?
- ▶ Genera **nuevos individuos** (descendientes) combinando características de sus progenitores.



## OPERADORES GENÉTICOS - CRUZA

- ▶ Se asigna un número aleatorio entre 0 y 1 a cada individuo seleccionado.

$$I_1 = 100011101 \rightarrow r_1 = 0.39$$

$$I_2 = 001011010 \rightarrow r_2 = 0.92$$

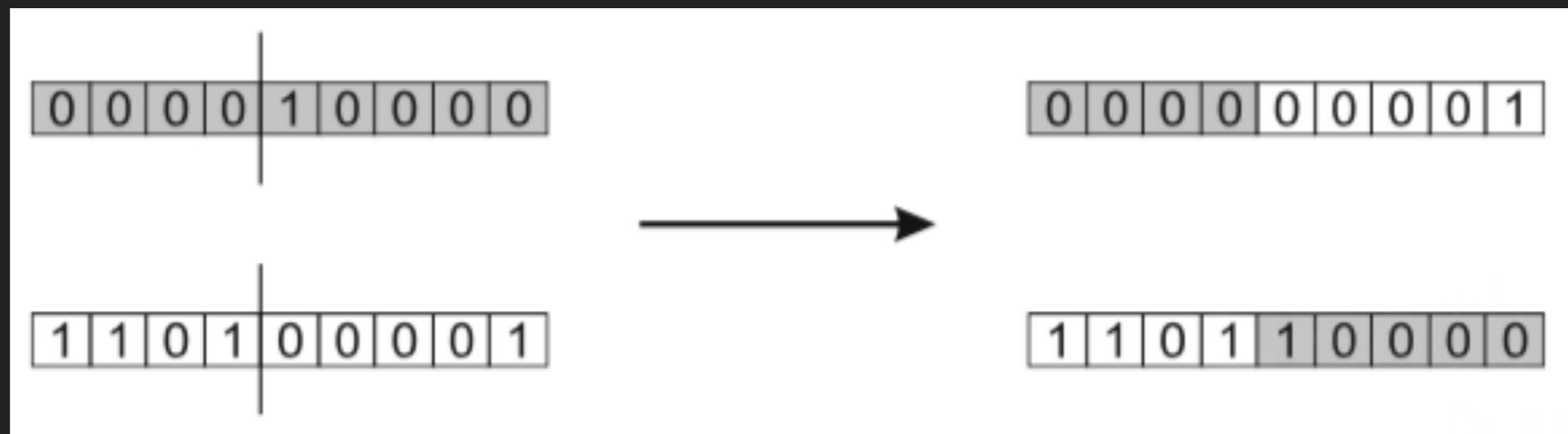
$$I_3 = 001011010 \rightarrow r_3 = 0.06$$

...

$$I_N = 110111010 \rightarrow r_N = 0.27$$

- ▶ Se comparan los  $r_i$  con una probabilidad de cruza preestablecida en el algoritmo.  
Ejemplo:  $p_{\text{cruza}} = 0.8$
- ▶ Si  $r_i < p_{\text{cruza}}$  entonces se produce la cruza entre los individuos que cumplen con esa condición.

# CRUZA MONOPUNTO



# CRUZA MULTIPUNTO



## CRUZA UNIFORME

- Para cada posición (gen) se decide aleatoriamente si se intercambian las posiciones.



## OPERADORES GENÉTICOS - SELECCIÓN

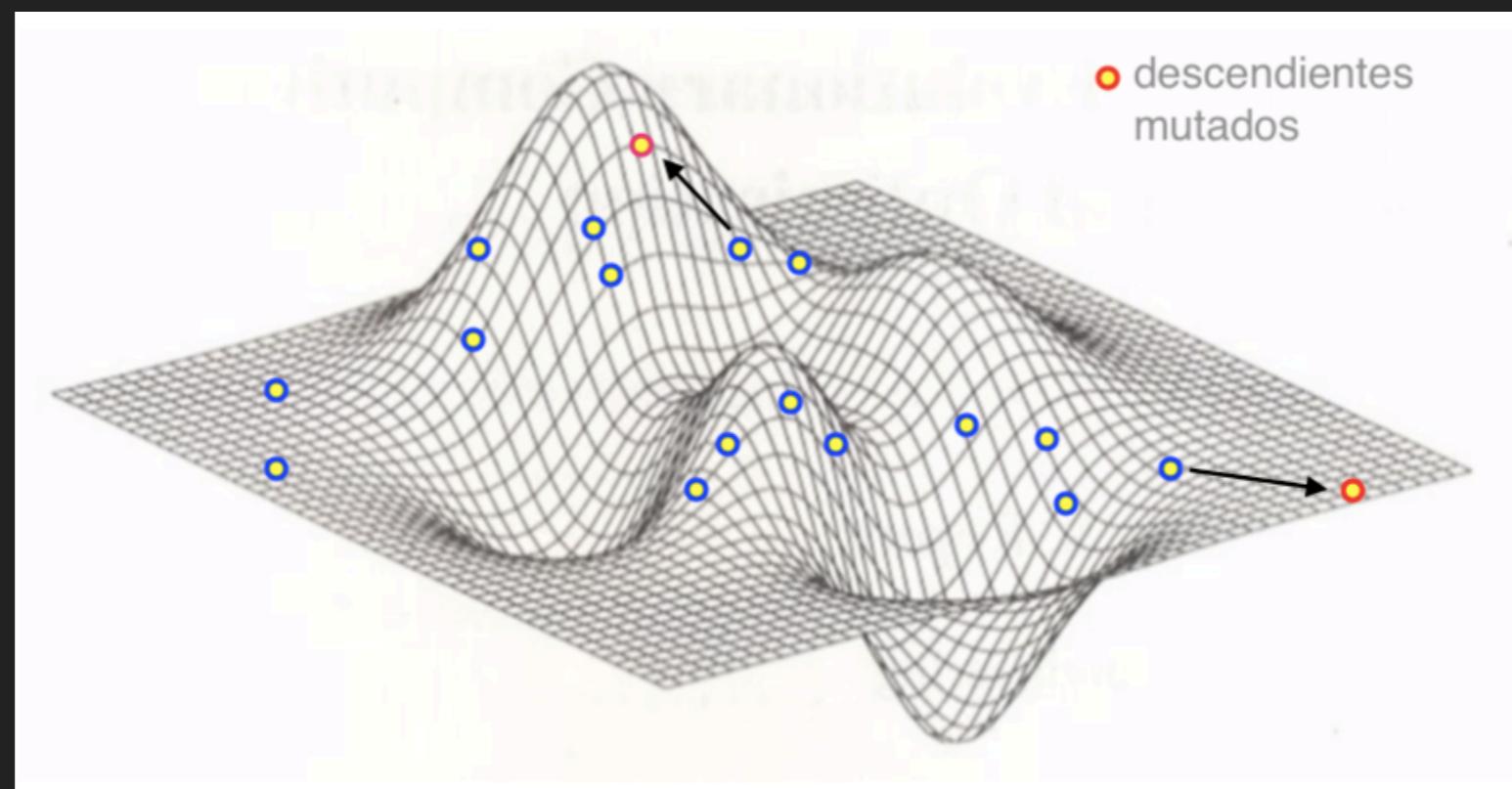
✓ Selección

✓ Cruza

✓ Mutación

## OPERADORES GENÉTICOS - SELECCIÓN

- ▶ ¿Para qué sirve la **Mutación**?
- ▶ Introduce **variabilidad** en la población para evitar la convergencia prematura.



## MUTACIÓN

- ▶ Consiste en alterar un alelo ubicado en un gen elegido aleatoriamente



- ▶ Se emplea una probabilidad **Pm** muy baja generalmente.
- ▶ Para cada gen de un individuo se genera un número aleatorio **Rm** en el intervalo  $(0, 1)$  y se compara con **Pm**
- ▶ Si  $Rm < Pm$  se muta el alelo correspondiente al gen evaluado.

## PROPÓSITO DE LA MUTACIÓN



- ▶ La mutación tiene como objetivo principal introducir diversidad genética en la población.
- ▶ Sin mutación, los algoritmos genéticos corren el riesgo de converger prematuramente hacia una solución subóptima, ya que podría no haber suficiente exploración del espacio de búsqueda.

## TASA (O PROBABILIDAD) DE MUTACIÓN

- ▶ La tasa de mutación (**Pm**) es un parámetro crítico que debe ajustarse con cuidado.
- ▶ Pm demasiado alta => búsqueda excesivamente aleatoria y **dificulta la convergencia** hacia soluciones óptimas.
- ▶ Pm demasiado baja => puede no introducir suficiente **variabilidad** en la población y el algoritmo podría quedarse estancado en mínimos locales.
- ▶ Se utiliza por lo general **Pm = 0.1 o inferior**



## COMO REEMPLAZAR A LOS PROGENITORES

- ▶ En el **modelo generacional**, la nueva población de descendientes sustituye completamente a la anterior.
- ▶ El no reemplazar al mejor cromosoma (o a los mejores cromosomas) de la población se conoce como **Elitismo**
- ▶ El **Elitismo** se utiliza en los modelos generacionales con la finalidad de no perder la mejor (o las mejores) solución/es encontrada/s.

# EJEMPLO

.UBAfiuba   
FACULTAD DE INGENIERÍA

## EJEMPLO (MAXIMIZAR $X^2$ Y SELECCIÓN POR RULETA)

- ▶ Maximizar la función  $f(x) = x^2$  en el intervalo  $[0, 31]$
- ▶ Representación (binario):  $2^5 = 32 \rightarrow 5$  dígitos, 0 (00000) a 31 (11111)
- ▶  $1x2^4 + 1x2^3 + 1x2^2 + 1x2^1 + 1x2^0 = 31$
- ▶ Suponemos una población inicial  $N = 4$
- ▶ Población inicial aleatoria:

I1 = 01101

I2 = 01000

I3 = 11000

I4 = 10011

## EJEMPLO (MAXIMIZAR X<sup>2</sup> Y SELECCIÓN POR RULETA)

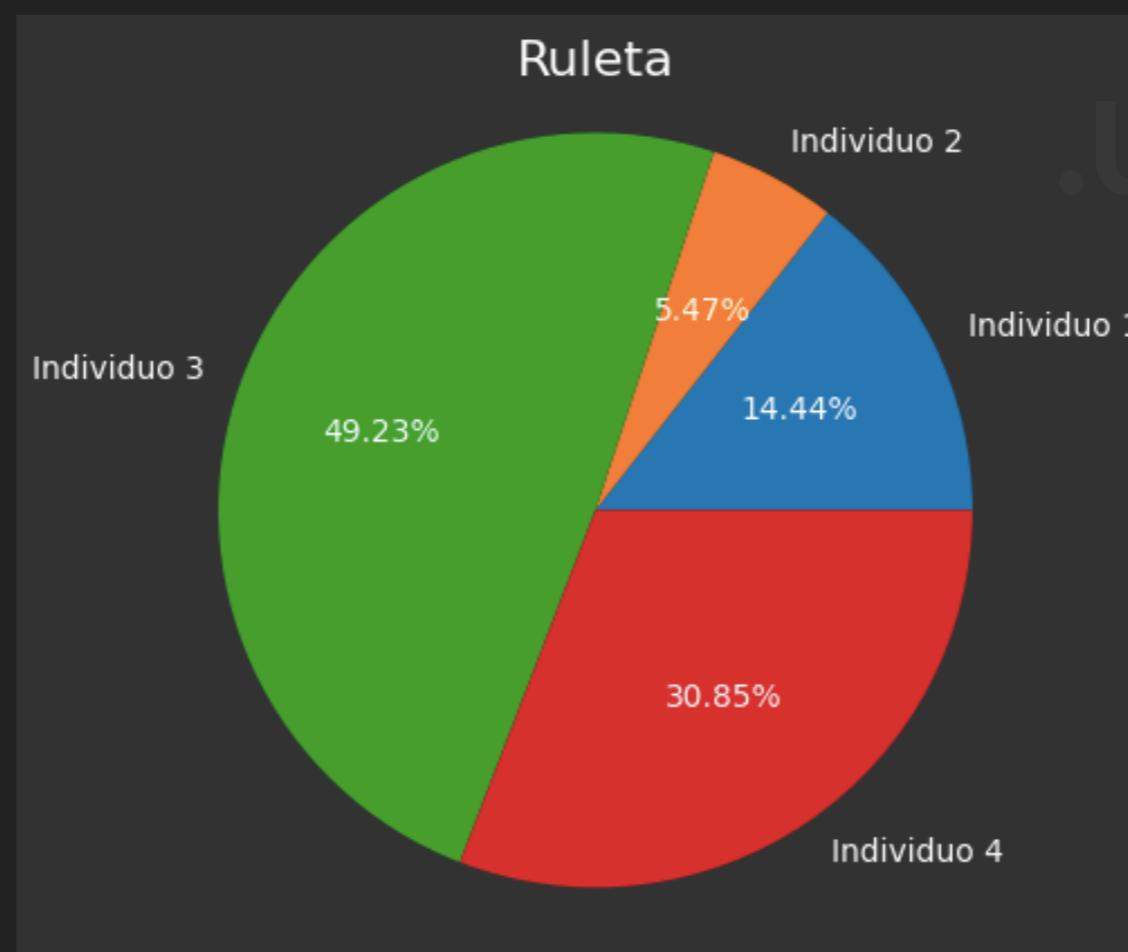
### ► Iteración 1

Individuo	Población	X	f(x) = x <sup>2</sup>	Aptitud	Probabilidad acumulada
1	01101	13	169	169/1170≈0.1444	0.1444
2	01000	8	64	64/1170≈0.0547	0.1991
3	11000	24	576	576/1170≈0.4923	0.6914
4	10011	19	361	361/1170≈0.3085	1.0000

►  $\sum f(x) = 169 + 64 + 576 + 361 = 1170$

## EJEMPLO (MAXIMIZAR $X^2$ Y SELECCIÓN POR RULETA)

Individuo	Población	X	$f(x) = x^2$	Aptitud	Probabilidad acumulada
1	01101	13	169	$169/1170 \approx 0.1444$	0.1444
2	01000	8	64	$64/1170 \approx 0.0547$	0.1991
3	11000	24	576	$576/1170 \approx 0.4923$	0.6914
4	10011	19	361	$361/1170 \approx 0.3085$	1.0000



## EJEMPLO (MAXIMIZAR $X^2$ Y SELECCIÓN POR RULETA)

- ▶ Por cada individuo se genera un número aleatorio con probabilidad uniforme.
- ▶  $[r_1 \ r_2 \ r_3 \ r_4] = [0.58 \ 0.84 \ 0.11 \ 0.43]$
- ▶ 4 lanzamientos de ruleta

Individuo	Población	X	$f(x) = x^2$	Aptitud	Probabilidad acumulada
1	01101	13	169	$169/1170 \approx 0.1444$	0.1444
2	01000	8	64	$64/1170 \approx 0.0547$	0.1991
3	11000	24	576	$576/1170 \approx 0.4923$	0.6914
4	10011	19	361	$361/1170 \approx 0.3085$	1.0000

## EJEMPLO (MAXIMIZAR X<sup>2</sup> Y SELECCIÓN POR RULETA)

- ▶  $[r_1 \ r_2 \ r_3 \ r_4] = [0.58 \ 0.84 \ 0.11 \ 0.43]$
- ▶  $0.1991 < r_1 = 0.58 < 0.6914 \rightarrow$  se selecciona a  $I_3$
- ▶  $0.6914 < r_2 = 0.84 < 1.000 \rightarrow$  se selecciona a  $I_4$
- ▶  $0.0 < r_3 = 0.11 < 0.1444 \rightarrow$  se selecciona a  $I_1$
- ▶  $0.1991 < r_4 = 0.43 < 0.6914 \rightarrow$  se selecciona a  $I_3$

Individuo	Población	X	$f(x) = x^2$	Aptitud	Probabilidad acumulada
1	01101	13	169	$169/1170 \approx 0.1444$	0.1444
2	01000	8	64	$64/1170 \approx 0.0547$	0.1991
3	11000	24	576	$576/1170 \approx 0.4923$	0.6914
4	10011	19	361	$361/1170 \approx 0.3085$	1.0000

## EJEMPLO (MAXIMIZAR $X^2$ Y SELECCIÓN POR RULETA)

- ▶ Progenitores:

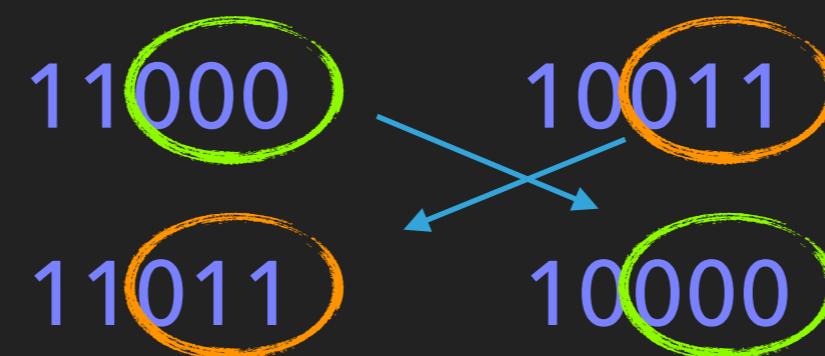
I3 con I4 (11000 con 10011)

I1 con I3 (01101 con 11000)

- ▶ Suponemos cruce monopunto luego del gen 2

Progenitores (I3 e I4) 11000

Descendientes



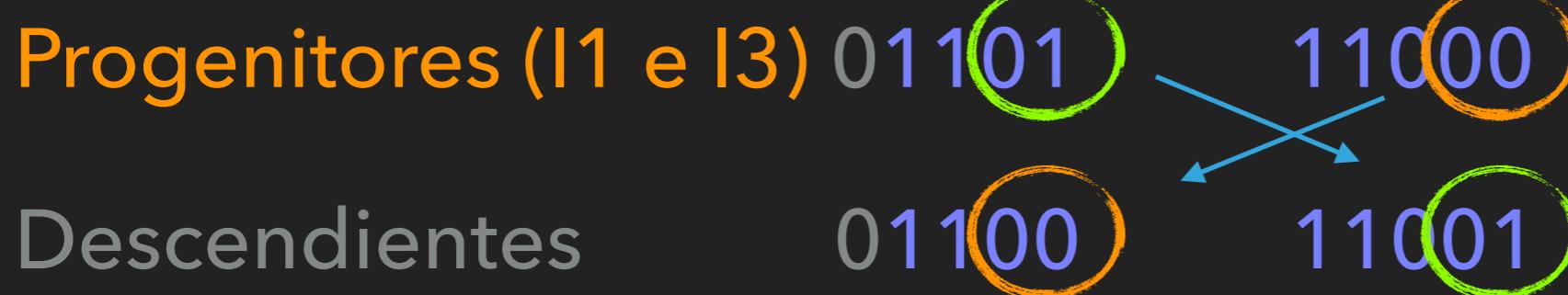
## EJEMPLO (MAXIMIZAR $X^2$ Y SELECCIÓN POR RULETA)

- ▶ Progenitores:

I3 con I4 (11000 con 10011)

I1 con I3 (01101 con 11000)

- ▶ Suponemos cruza monopunto luego del gen 3

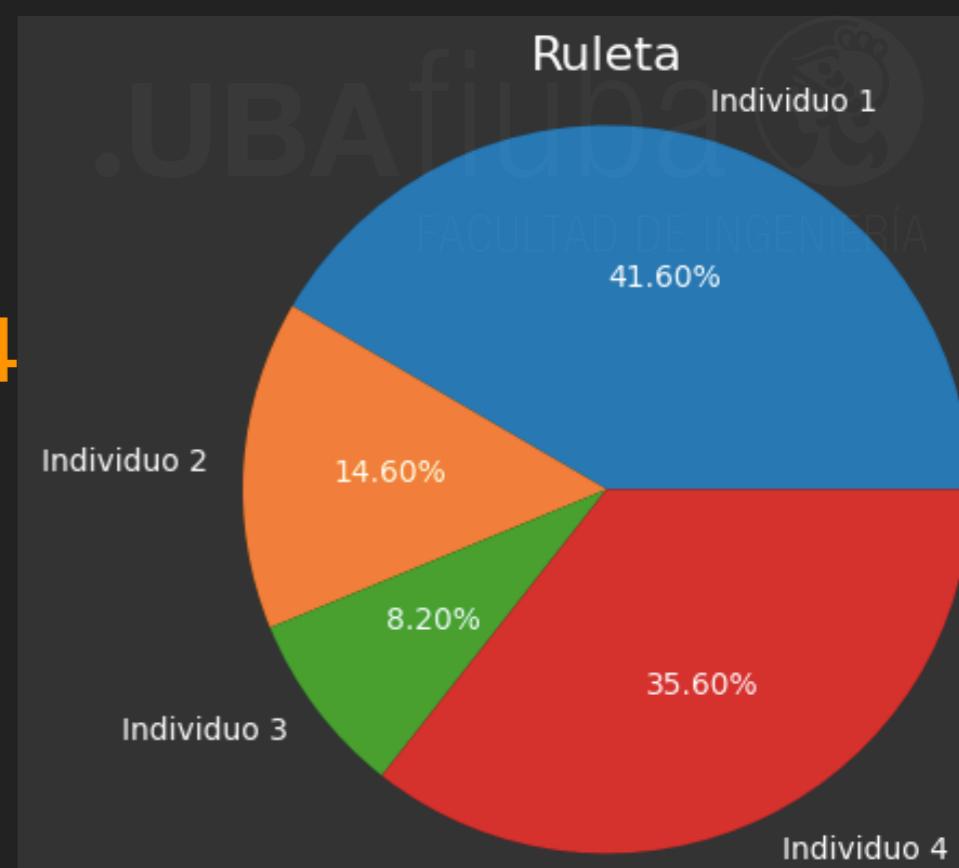


## EJEMPLO (MAXIMIZAR $X^2$ Y SELECCIÓN POR RULETA)

Individuo	Población	X	$f(x) = x^2$	Aptitud	Probabilidad acumulada
1'	11011	27	729	729/1754≈0.416	0.416
2'	10000	16	256	256/1754≈0.146	0.562
3'	01100	12	144	144/1754≈0.082	0.644
4'	11001	25	625	625/1754≈0.356	1.0000

▶ Iteración 2

▶  $\sum f(x) = 729 + 256 + 144 + 625 = 1754$



## EJEMPLO (MAXIMIZAR $X^2$ Y SELECCIÓN POR RULETA)

- ▶ Para cada individuo  $I'$  se genera un numero aleatorio con probabilidad uniforme.
- ▶  $[r_1 \ r_2 \ r_3 \ r_4] = [0.62 \ 0.7 \ 0.47 \ 0.79]$
- ▶ ¿Cuales serán los individuos seleccionados?

Individuo	Población	X	$f(x) = x^2$	Aptitud	Probabilidad acumulada
1'	11011	27	729	$729/1754 \approx 0.416$	0.416
2'	10000	16	256	$256/1754 \approx 0.146$	0.562
3'	01100	12	144	$144/1754 \approx 0.082$	0.644
4'	11001	25	625	$625/1754 \approx 0.356$	1.0000

## EJEMPLO (MAXIMIZAR X<sup>2</sup> Y SELECCIÓN POR RULETA)

- ▶  $[r_1' \ r_2' \ r_3' \ r_4'] = [0.62 \ 0.7 \ 0.47 \ 0.79]$
- ▶  $0.562 < r_1' = 0.62 < 0.644 \rightarrow$  se selecciona a  $I_{3'}$
- ▶  $0.644 < r_2' = 0.7 < 1.000 \rightarrow$  se selecciona a  $I_{4'}$
- ▶  $0.416 < r_3' = 0.47 < 0.562 \rightarrow$  se selecciona a  $I_{2'}$
- ▶  $0.644 < r_4' = 0.79 < 1.000 \rightarrow$  se selecciona a  $I_{4'}$

Individuo	Población	X	$f(x) = x^2$	Aptitud	Probabilidad acumulada
1'	11011	27	729	$729/1754 \approx 0.416$	0.416
2'	10000	16	256	$256/1754 \approx 0.146$	0.562
3'	01100	12	144	$144/1754 \approx 0.082$	0.644
4'	11001	25	625	$625/1754 \approx 0.356$	1.0000

## EJEMPLO (MAXIMIZAR $X^2$ Y SELECCIÓN POR RULETA)

- ▶ Progenitores:

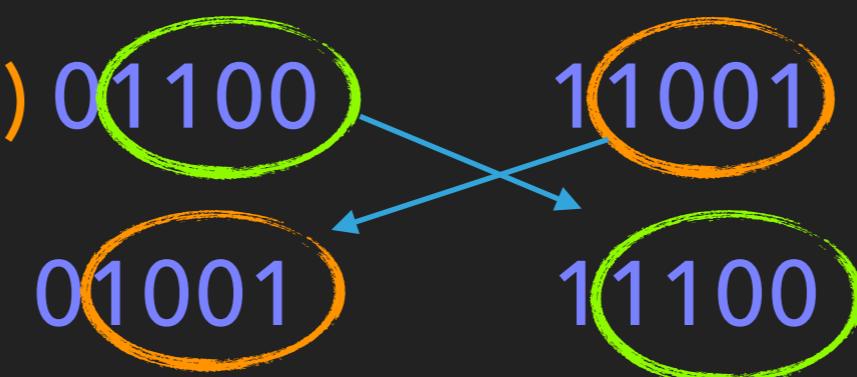
I3' con I4' (01100 con 11001)

I2' con I4' (10000 con 11001)

- ▶ Suponemos cruce monopunto luego del gen 1

Progenitores (I3' e I4') 01100

Descendientes



## EJEMPLO (MAXIMIZAR $X^2$ Y SELECCIÓN POR RULETA)

- ▶ Progenitores:

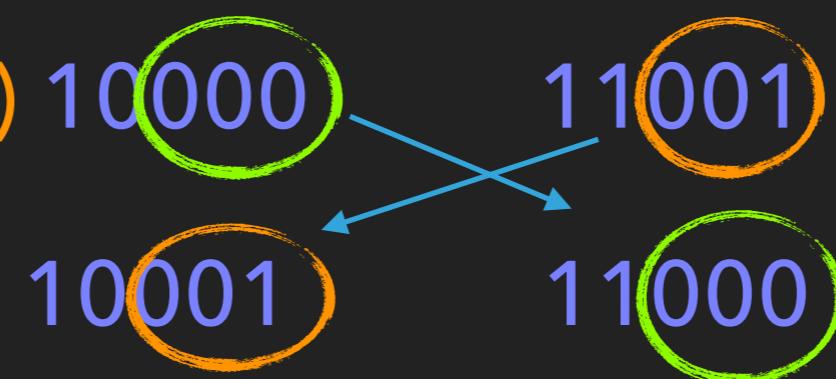
I3' con I4' (01100 con 11001)

I2' con I4' (10000 con 11001)

- ▶ Suponemos cruce monopunto luego del gen 2

Progenitores (I2' e I4') 10000

Descendientes



## EJEMPLO (MAXIMIZAR X<sup>2</sup> Y SELECCIÓN POR RULETA)

- ▶ Nuevos progenitores:

$$I_1'' = 01001, I_2'' = 11100, I_3'' = 10001, I_4'' = 11000$$

Individuo	Población	X	f(x) = x <sup>2</sup>	Aptitud	Probabilidad acumulada
1''	01001	9	81	81/1730≈0.047	0.047
2''	11100	28	784	784/1730≈0.453	0.500
3''	10001	17	289	289/1730≈0.167	0.667
4''	11000	24	576	576/1730≈0.333	1.0000

## EJEMPLO (MAXIMIZAR $X^2$ Y SELECCIÓN POR RULETA)

- ▶ Continuar iterando (producir nuevas generaciones) hasta alcanzar el óptimo exacto o aproximado....

# PRECISIÓN DE LOS INDIVIDUOS CON DECIMALES



## COMO ESTABLECER PRECISIÓN CON DECIMALES EN UN INDIVIDUO BINARIO.

- ▶ Supongamos que debemos optimizar la siguiente función objetivo:

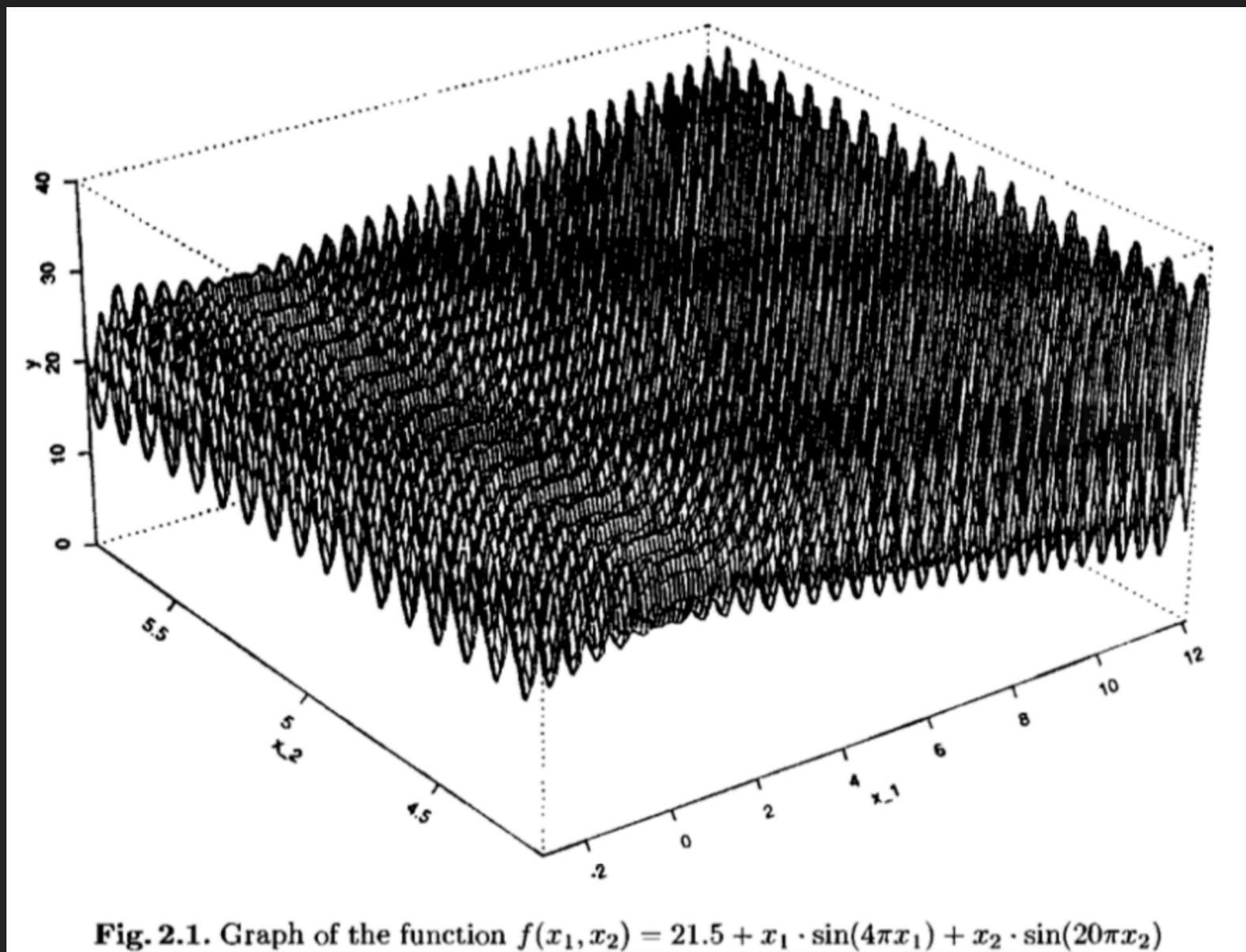
$$f(x_1, x_2) = 21.5 + x_1 \cdot \sin(4\pi x_1) + x_2 \cdot \sin(20\pi x_2)$$

- ▶ Donde:

$$-3.0 \leq x_1 \leq 12.1 \text{ and } 4.1 \leq x_2 \leq 5.8.$$

## COMO ESTABLECER PRECISIÓN CON DECIMALES EN UN INDIVIDUO BINARIO.

- ▶ Gráficamente la función es:



## COMO ESTABLECER PRECISIÓN CON DECIMALES EN UN INDIVIDUO BINARIO.

- ▶ Supongamos que queremos definir una precisión de **4 decimales**.
- ▶ Como  $x_1$  tiene límites superior e inferior de:

$$[-3.0, 12.1]$$

- ▶ La diferencia de estos límites me indica el rango de  $x_1$ , es decir, **15.1**
- ▶ Por tanto el rango debería ser dividido en al menos  **$15.1 * 10_000$**  rangos de igual tamaño.
- ▶ En binario significa que son 18 bits los necesarios:  $2^{17} < 151000 \leq 2^{18}$
- ▶  $2^{17} = 131_000, 2^{18} = 262_144 \Rightarrow dígitos_{x_1} = 18 \text{ bits}$

## COMO ESTABLECER PRECISIÓN CON DECIMALES EN UN INDIVIDUO BINARIO.

- ▶ Veamos para  $x_2$ :
- ▶ Tenemos:  $[4.1, 5.8]$  o sea, el rango es 1.7
- ▶ Se requiere por tanto:  $1.7 \cdot 10000$
- ▶ Esto es:  $2^{14} < 17000 \leq 2^{15}$ , es decir,  $dígitos_{x_2} = 15$  bits.

## COMO ESTABLECER PRECISIÓN CON DECIMALES EN UN INDIVIDUO BINARIO.

- ▶ Por tanto el tamaño del cromosoma será:  $m = 18+15 = 33$
- ▶ 18 bits para  $x_1$  y 15 bits para  $x_2$
- ▶ Un cromosoma aleatorio podría ser el siguiente:

(01000100101101000011110010100010)

- ▶ Si tomo los primeros 18 bits ( $x_1$ ):

$$x_1 = -3.0 + \text{decimal}(010001001011010000_2) \cdot \frac{12.1 - (-3.0)}{2^{18}-1} = -3.0 + \\ 70352 \cdot \frac{15.1}{262143} = -3.0 + 4.052426 = 1.052426.$$

## COMO ESTABLECER PRECISIÓN CON DECIMALES EN UN INDIVIDUO BINARIO.

- ▶ Si tomo los 15 bits restantes ( $x_2$ ):

$$x_2 = 4.1 + \text{decimal}(111110010100010_2) \cdot \frac{5.8 - 4.1}{2^{15} - 1} = 4.1 + 31906 \cdot \frac{1.7}{32767} =$$

$$4.1 + 1.655330 = 5.755330$$

- ▶ El par  $x_1, x_2$ :  $\langle x_1, x_2 \rangle = \langle 1.052426, 5.755330 \rangle$
- ▶ La función evaluada para el par  $x_1, x_2$  es:

$$f(1.052426, 5.755330) = 20.252640$$

## EJEMPLO 1

- ▶ Máximo de  $x^2$  en  $[0, 31] \in \mathbb{Z}$
- ▶ El número total de valores posibles es:

$$N = 31 - 0 + 1 = 32$$

- ▶ Para representar 32 valores distintos en binario se necesita:  
 $\text{digitos} = \lceil \log_2(N) \rceil = \lceil \log_2(32) \rceil = 5 \text{ bits}$
- ▶ Resultado: 5 bits para representar x.

## EJEMPLO 2

- ▶ Mínimo de  $x^2$  en  $[-31, 31] \in \mathbb{Z}$
- ▶ El número total de valores posibles es:  
$$N = 31 - (-31) + 1 = 63$$
- ▶ Para representar 63 valores distintos en binario se necesita:  
$$\text{digitos} = \lceil \log_2(N) \rceil = \lceil \log_2(63) \rceil = 6 \text{ bits}$$
- ▶ Resultado: 6 bits para representar x.

## EJEMPLO 2

- ▶ Mínimo de  $x^2$  en  $[-31, 31] \in \mathbb{R}$  (1 decimal de precisión)
- ▶ El número total de valores posibles es:  
$$N = [(31 - (-31)) / 0.1] + 1 = 62 / 0.1 + 1 = 621$$
- ▶ Para representar 621 valores distintos en binario se necesita:  
$$\text{dígitos} = \lceil \log_2(N) \rceil = \lceil \log_2(621) \rceil = 10 \text{ bits}$$
- ▶ Resultado: 10 bits para representar x.

## REFERENCIAS BIBLIOGRÁFICAS Y WEB

- ▶ Eiben, A. E., & Smith, J. E. (2015). *Introduction to evolutionary computing*. Springer-Verlag Berlin Heidelberg.
- ▶ Holland, J. H. (1992). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press.
- ▶ Mitchell, M. (1998). *An introduction to genetic algorithms*. MIT press.
- ▶ Michalewicz Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs*. Springer Verlag.