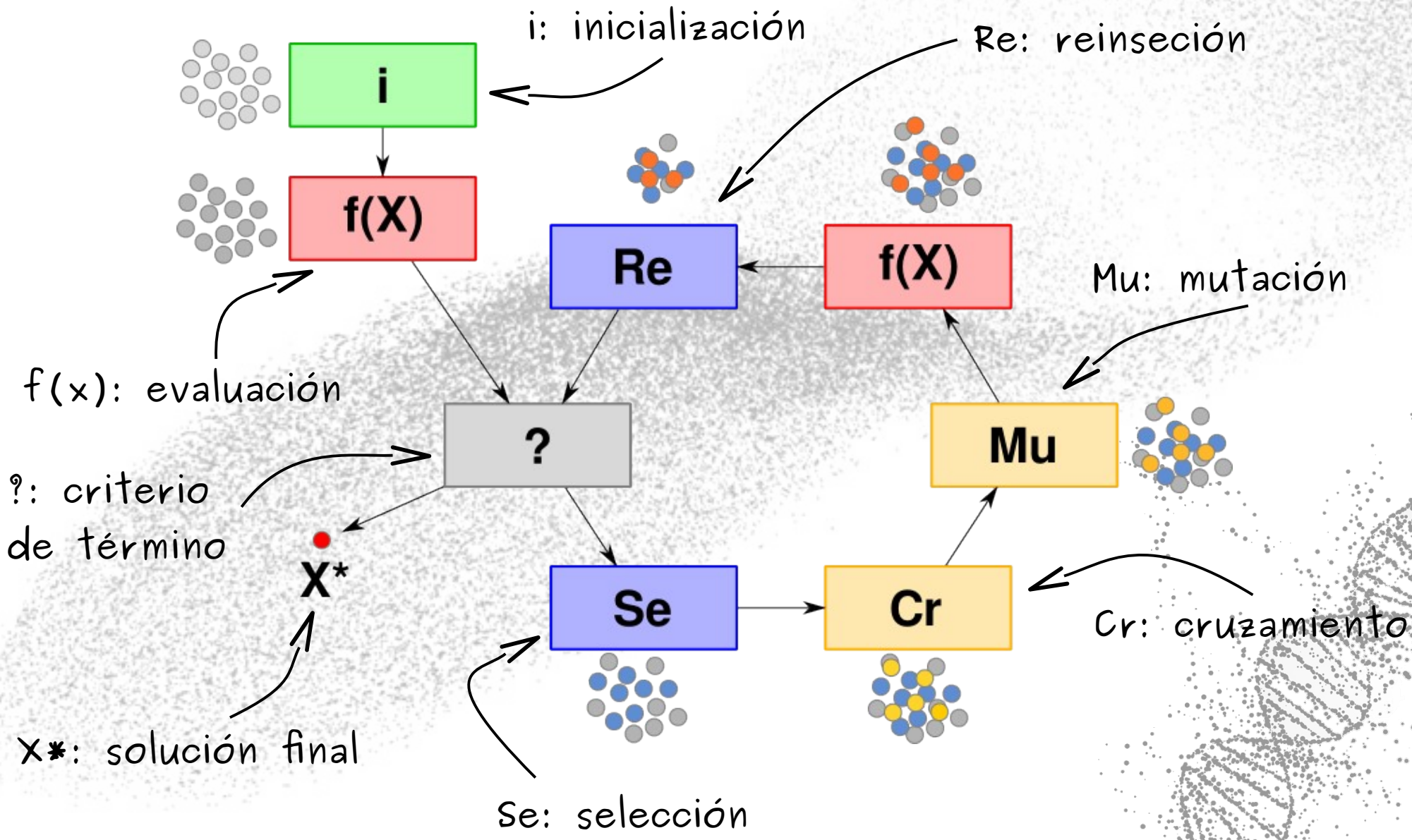
Algoritmos de optimización bioinspirados

Algoritmos Evolutivos, cómo funcionan

Inteligencia Artificial
INFO257

Profesor: Jorge Maturana
jorge.maturana@inf.uach.cl

Algoritmo Evolutivo



Cómo funciona un Algoritmo Evolutivo?

- Ejemplo: Optimización de una función en 2D

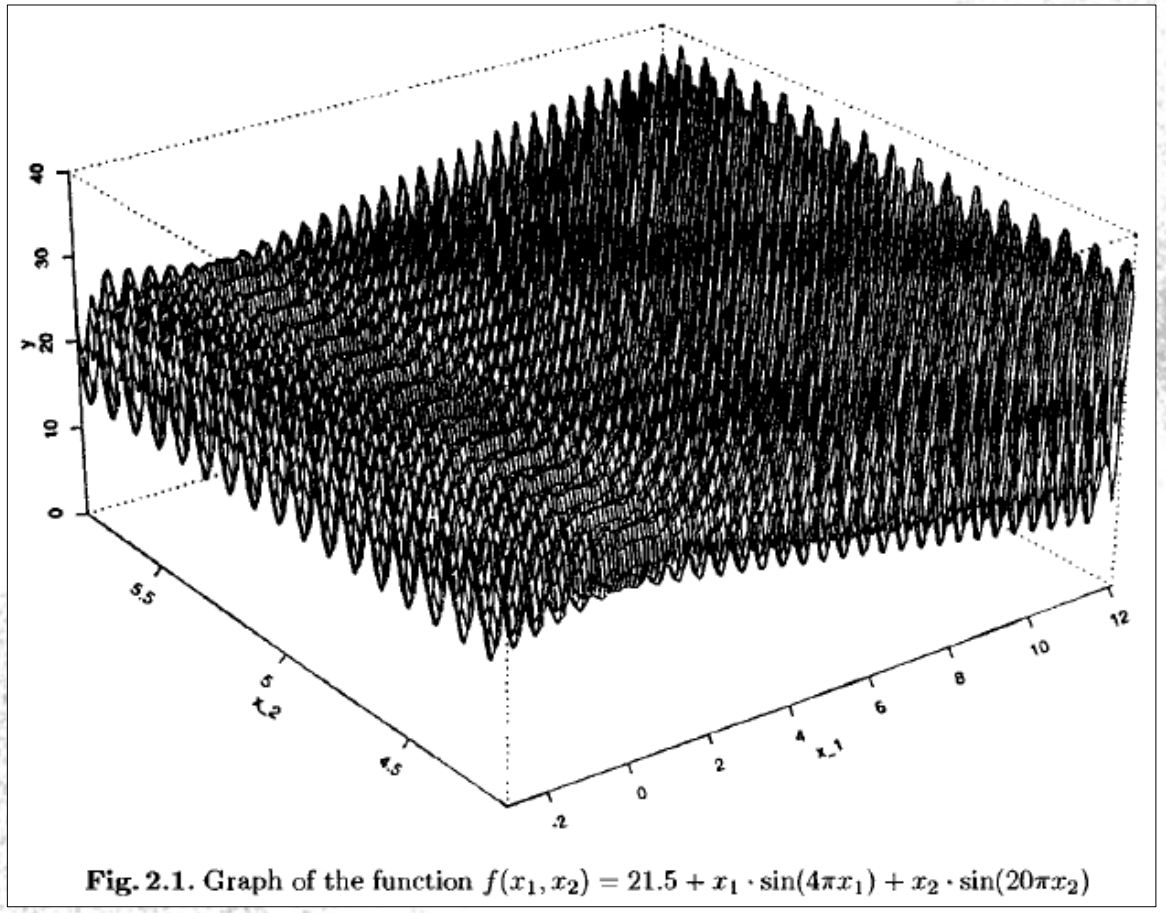


Fig. 2.1. Graph of the function $f(x_1, x_2) = 21.5 + x_1 \cdot \sin(4\pi x_1) + x_2 \cdot \sin(20\pi x_2)$

(Michalewicz, Ch.2)

Representación (encoding)

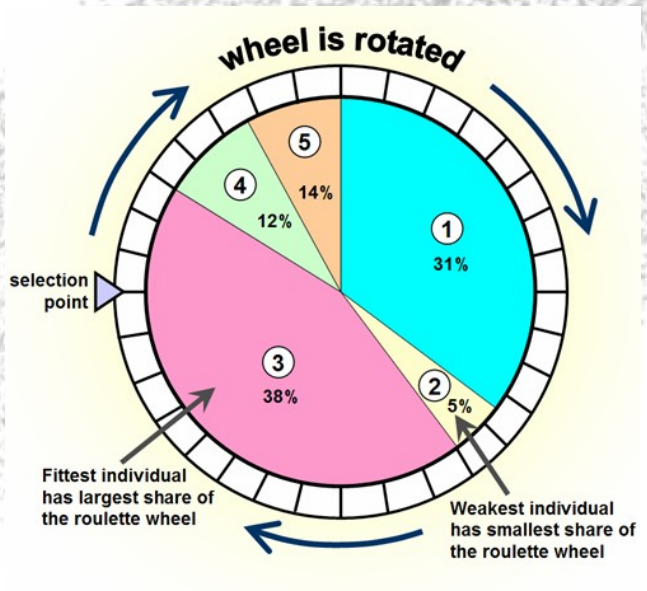
- Usando Algoritmos Genéticos (AG), propuestos por John Holland (1975)
- Rango de variables:

$$3.0 \leqslant x_1 \leqslant 12.1, 4.1 \leqslant x_2 \leqslant 5.8$$

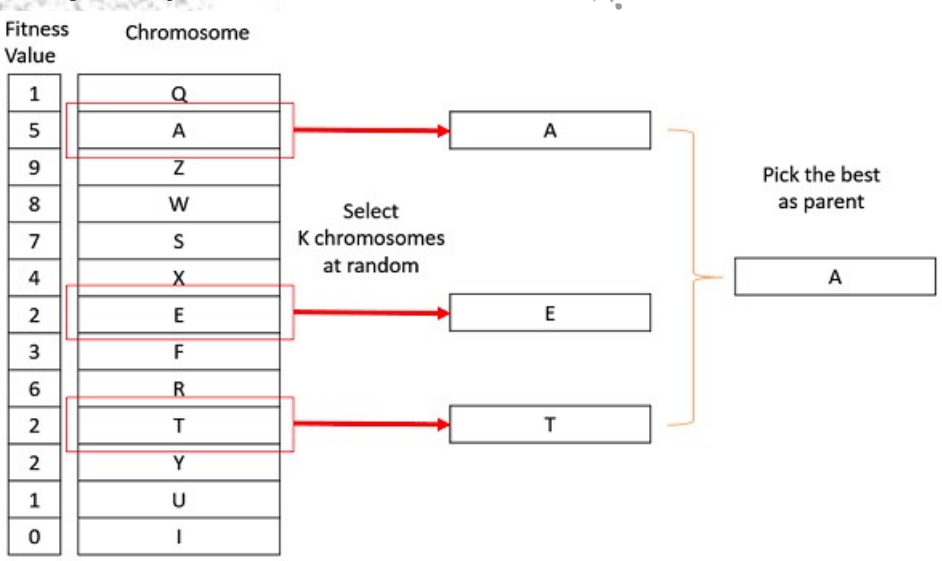
- Representación clásica de los Algoritmos Genéicos: strings binarios
  - Cada variable será representada por un string de bits
  - 000...0 corresponde al valor mínimo, 111...1 al valor máximo
- Los Algoritmos Evolutivos (Clase más general) admiten **otras representaciones**:
  - **Reales**
  - **Permutaciones**
  - **Árboles**
  - **Cualquier otra estructura de datos, variable o no**
- Se determina un tamaño de población (e.g., 20) y se genera una población de individuos inicializados al azar

Evaluación y selección

- Cada individuo es decodificado y evaluado: función de ***fitness***
- Se crea una nueva población de 20 individuos seleccionando con preferencia a los mejores
  - Idea: sólo los mejores tienen derecho a procrear
  - Método base: Roulette wheel
  - N.B. Algunos pueden quedar seleccionados más de una vez

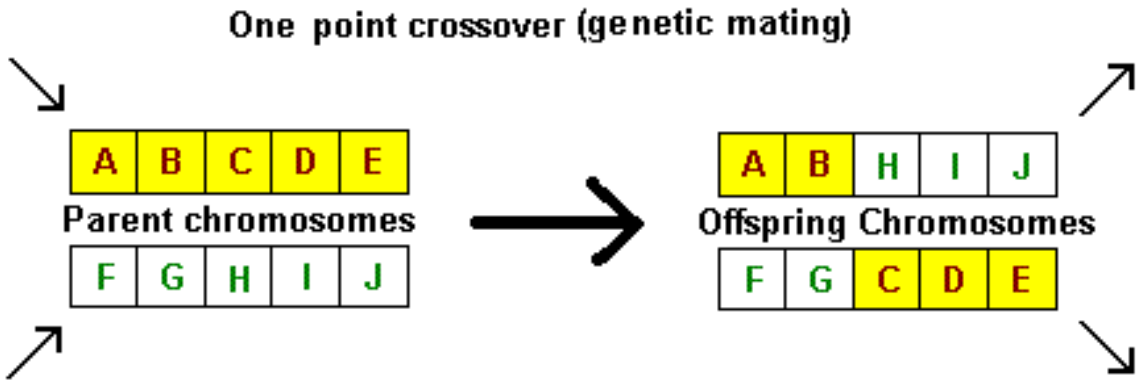


Mejor opción: **tournament selection**



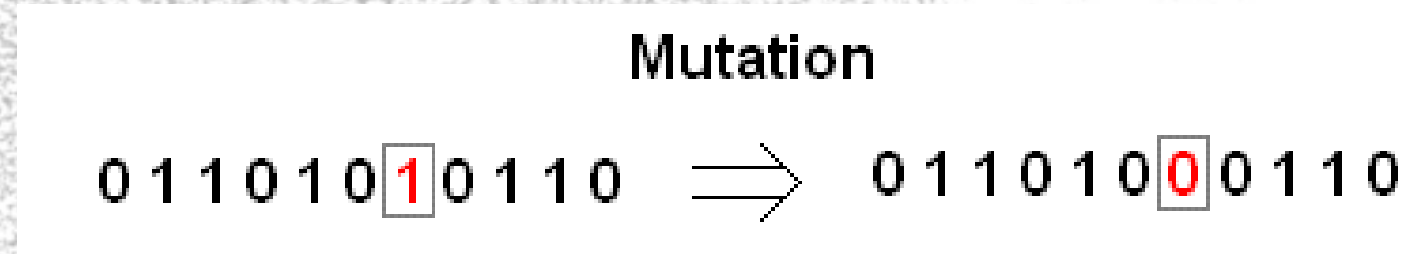
Cruzamiento

- Se elige con alguna probabilidad distintos individuos para mezclar sus cromosomas
  - Típicamente probabilidad entre 0.2 y 0.8
  - Una vez elegidos los dos ***padres***, elegir al azar un punto de corte e intercambiar las dos mitades, para producir dos ***hijos***
  - Este cruzamiento se conoce como “de un punto”
    - **Existen otros!,** dependiendo de la codificación y el problema
  - Los hijos reemplazan a los padres en la población (la secundaria)



Mutación

- Una vez cruzados, comienza la mutación
  - Cambio con probabilidad baja (~0.01)
  - Para la codificación de string de bits, se acostumbra a decidir bit a bit
  - Si un número al azar es inferior a la probabilidad de mutación, se hace un swap del bit (0→1 ó 1→0)



  - **Existen otras!**, dependiendo de la codificación y el problema

Reinserción

- En general existen dos estrategias de reinserción:
  - **Generacional**
    - Los individuos nuevos “pisan” a los antiguos
    - La población se renueva constantemente
    - Exploración relativamente alta
  - **Steady State**
    - Los individuos deben competir con sus padres
    - A medida que avanza la búsqueda, aumenta la explotación
    - Eventual atasco en la búsqueda

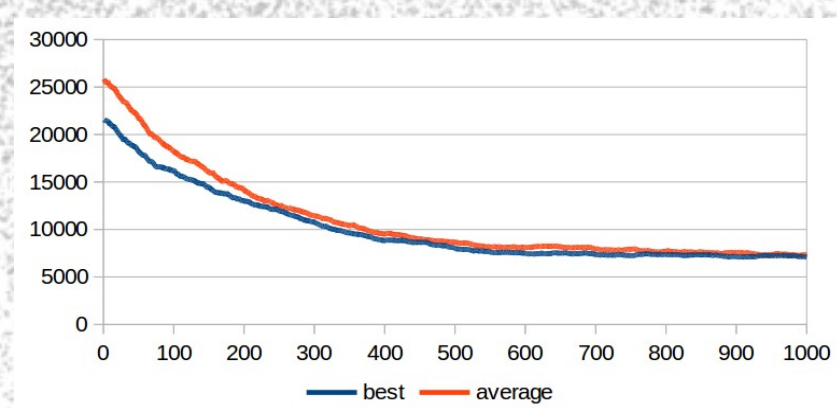
Exploración vs Explotación

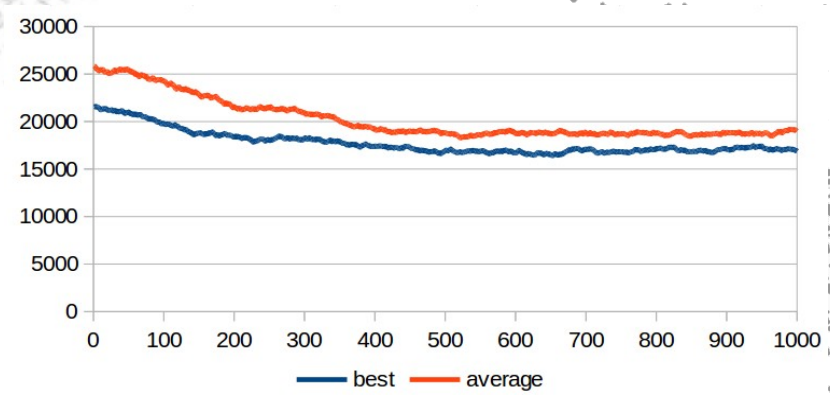
- Dos formas de buscar:

- Distintas estrategias (ambas necesarias)
  - Buscar ampliamente? → Explorar / diversificar
  - Refinar búsqueda? → Explotar / intensificar

Lidiando con el azar

- La aleatoriedad dificulta el debug y la evaluación del algoritmo
- Tips:
  - Fijar o registrar semilla aleatoria para reproducir errores
  - Evitar usar una misma semilla siempre: caso puntual
  - Usar un gráfico de convergencia:





Gráficas típicas de convergencia