

# ALGORÍTMICA

## 2012 – 2013

---

- **Parte I. Introducción a las Metaheurísticas**
  - Tema 1. Metaheurísticas: Introducción y Clasificación
- **Parte II. Métodos Basados en Trayectorias y Entornos**
  - Tema 2. Algoritmos de Búsqueda Local Básicos
  - Tema 3. Algoritmos de Enfriamiento Simulado
  - Tema 4. Algoritmos de Búsqueda Tabú
  - Tema 5. Métodos Basados en Trayectorias Múltiples I: Métodos Multiarranque Básicos y GRASP
  - Tema 6. Métodos Basados en Trayectorias Múltiples II: ILS y VNS
- **Parte III. Métodos Basados en Poblaciones**
  - **Tema 7. Algoritmos Genéticos**
- **Parte IV. Intensificación y Diversificación**
  - Tema 8. Estudio del Equilibrio entre Intensificación y Diversificación
- **Parte V. Metaheurísticas Híbridas: Poblaciones y Trayectorias**
  - Tema 9. Algoritmos Meméticos
  - Tema 10. *Scatter Search*
- **Parte VI. Paralelización de Metaheurísticas**
  - Tema 11. Metaheurísticas en Sistemas Descentralizados
- **Parte VII. Conclusiones**
  - Tema 12. Algunas Consideraciones sobre la Adaptación de Metaheurísticas a la Resolución de Problemas

# ALGORÍTMICA

## TEMA 7: ALGORITMOS GENÉTICOS

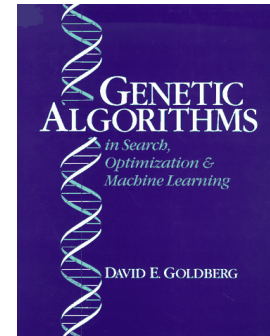
---

1. ALGORITMOS GENÉTICOS. INTRODUCCIÓN
2. ¿CÓMO SE CONSTRUYE?
3. SOBRE SU UTILIZACIÓN
4. ALGUNAS CUESTIONES: DIVERSIDAD, EXPLORACIÓN vs EXPLOTACIÓN
5. MODELOS: GENERACIONAL vs ESTACIONARIO
6. DOMINIOS DE APLICACIÓN
7. EJEMPLO: VIAJANTE DE COMERCIO
8. APLICACIONES

# BIBLIOGRAFÍA

---

**D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley, 1989.**



Clásico en AG

**Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs. Springer Verlag, 1996.**

**T. Bäck, D.B. Fogel, Z. Michalewicz, Handbook of Evolutionary Computation, Institute of Physics Publishers, 1997.**

**A.E. Eiben, J.E. Smith. Introduction to Evolutionary Computing. Springer, 2003.**

# 1. INTRODUCCIÓN A LOS ALGORITMOS GENÉTICOS

---

- **EVOLUCIÓN NATURAL. EVOLUCIÓN ARTIFICIAL**
- **¿QUÉ ES UN ALGORITMO GENÉTICO?**
- **LOS INGREDIENTES**
- **EL CICLO DE LA EVOLUCIÓN**
- **ESTRUCTURA DE UN ALGORITMO GENÉTICO**

# Evolución Natural (1)

---

**En la naturaleza, los procesos evolutivos ocurren cuando se satisfacen las siguientes condiciones:**

Una entidad o individuo tiene la habilidad de reproducirse

Hay una población de tales individuos que son capaces de reproducirse

Existe alguna variedad, diferencia, entre los individuos que se reproducen

Algunas diferencias en la habilidad para sobrevivir en el entorno están asociadas con esa variedad

# Evolución Natural (2)

---

**Los mecanismos que conducen esta evolución no son totalmente conocidos, pero sí algunas de sus características, que son ampliamente aceptadas:**

La evolución es un proceso que opera sobre los cromosomas más que sobre las estructuras de la vida que están codificadas en ellos

# Evolución Natural (3)

---

La selección natural es el enlace entre los cromosomas y la actuación de sus estructuras decodificadas.

El proceso de reproducción es el punto en el cual la evolución toma parte, actúa

La evolución biológica no tiene memoria

**Referencia: Darwin, C. (1859). *On the Origin of Species by Means of Natural Selection or the Preservations of Favoured Races in the Struggle for Life*. London: John Murray**

# Evolución Artificial (1)

---

## COMPUTACIÓN EVOLUTIVA

Está compuesta por modelos de evolución basados en poblaciones cuyos elementos representan soluciones a problemas

La simulación de este proceso en un ordenador resulta ser una técnica de optimización probabilística, que con frecuencia mejora a otros métodos clásicos en problemas difíciles



# Evolución Artificial (2)

---

Existen cuatro paradigmas básicos:

**Algoritmos Genéticos** que utilizan operadores genéticos sobre cromosomas

**Estrategias de Evolución** que enfatizan los cambios de comportamiento al nivel de los individuos

**Programación Evolutiva** que enfatizan los cambios de comportamiento al nivel de las especies

**Programación Genética** que evoluciona expresiones representadas como árboles

**Existen otros múltiples Modelos de Evolución de Poblaciones**

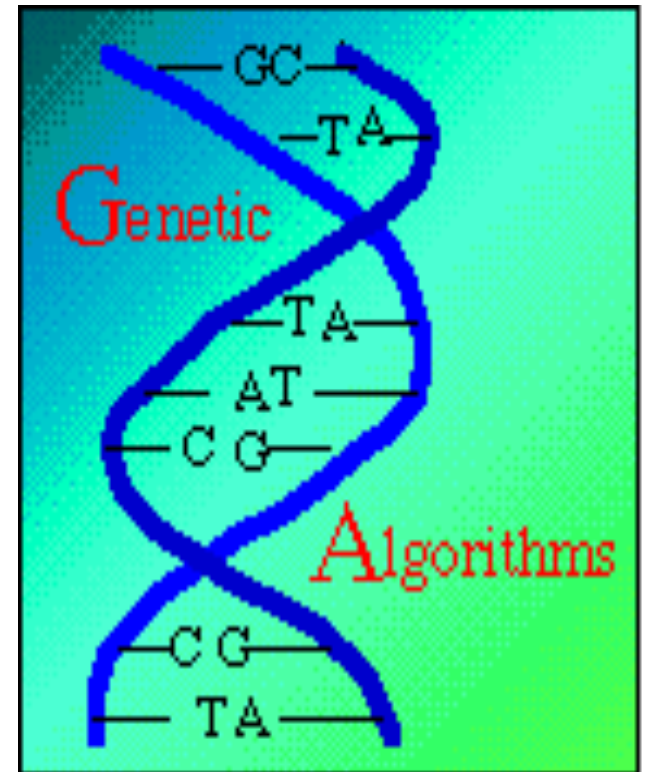
# ¿Qué es un Algoritmo Genético?

---

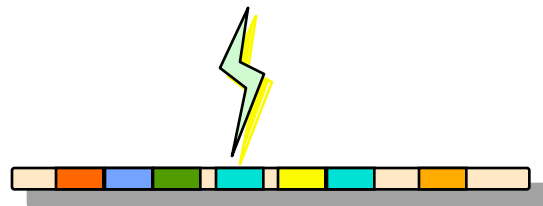
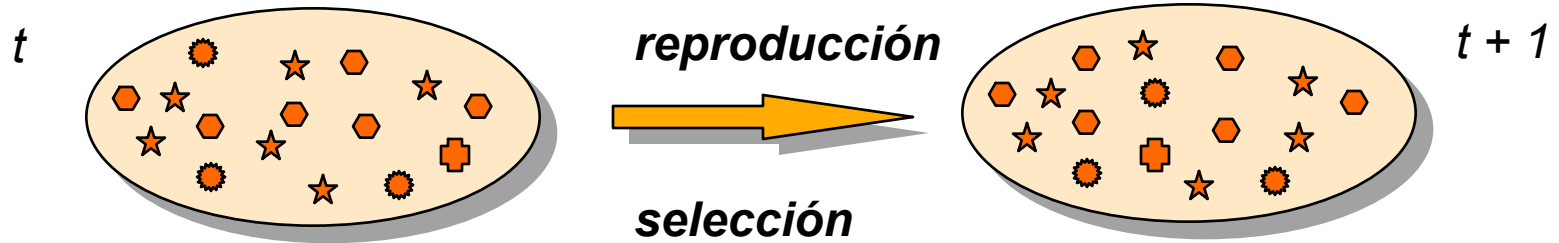
## Los Algoritmos Genéticos

son algoritmos de optimización  
búsqueda  
y aprendizaje  
inspirados en los procesos de

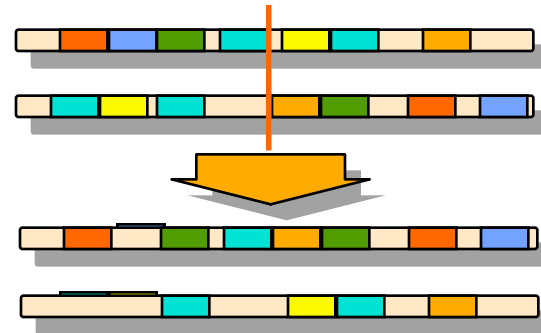
**Evolución Natural**  
**y**  
**Evolución Genética**



# Los Ingredientes



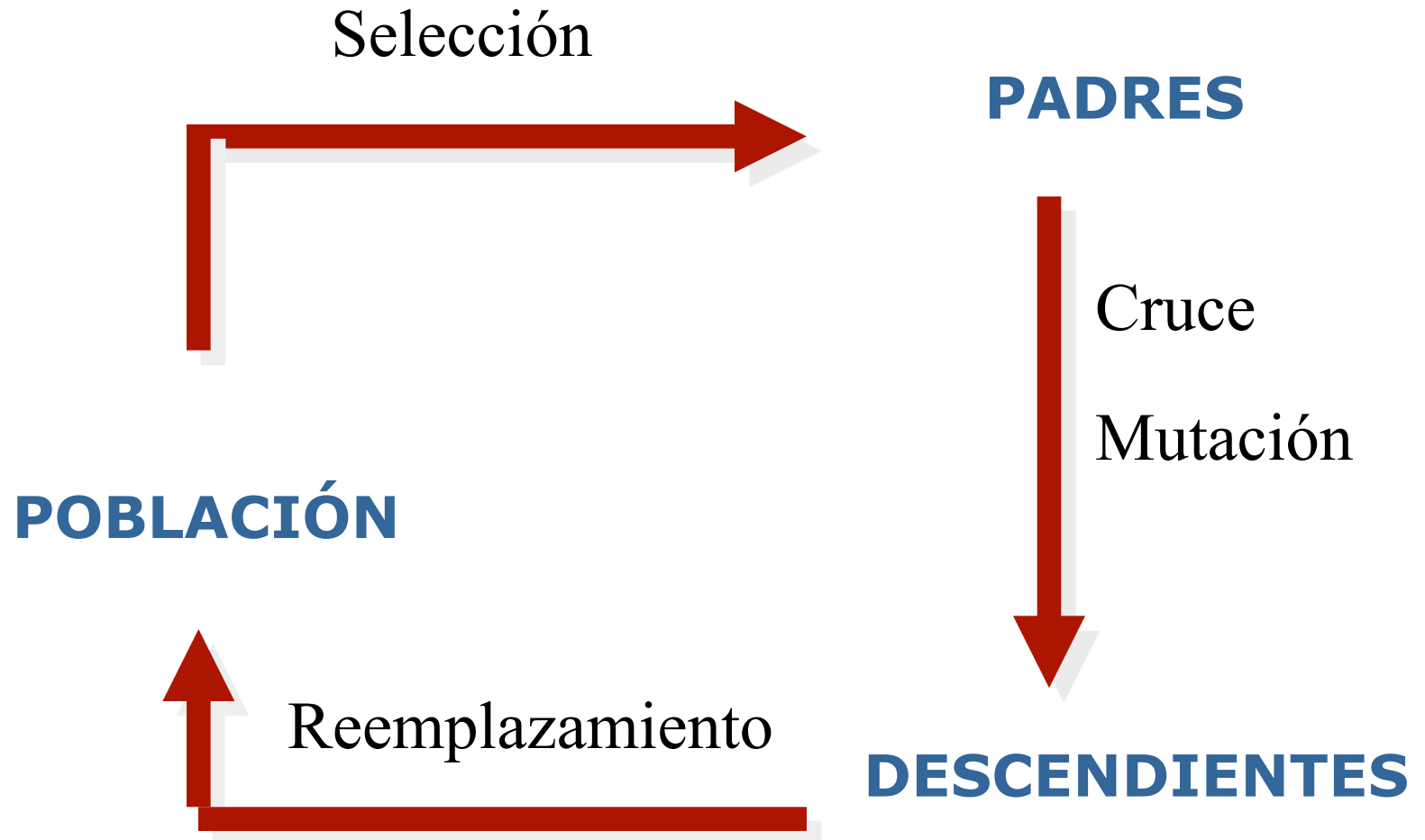
*mutación*



*cruce*

# El ciclo de la Evolución

---



# Estructura de un Algoritmo Genético

---

## Procedimiento Algoritmo Genético

Inicio (1)

$t = 0;$

inicializar  $P(t);$

evaluar  $P(t);$

Mientras (no se cumpla la condición de parada) hacer

Inicio(2)

$t = t + 1$

seleccionar  $P(t)$  desde  $P(t-1)$

recombinar  $P(t)$

mutación  $P(t)$

evaluar  $P(t)$

Final(2)

Final(1)

## 2. ¿CÓMO SE CONSTRUYE UN AG?

---

### Los pasos para construir un Algoritmo Genético

- Diseñar una representación
- Decidir cómo inicializar una población
- Diseñar una correspondencia entre genotipo y fenotipo
- Diseñar una forma de evaluar un individuo
- Diseñar un operador de mutación adecuado
- Diseñar un operador de cruce adecuado
- Decidir cómo seleccionar los individuos para ser padres
- Decidir cómo reemplazar a los individuos
- **Decidir la condición de parada**

DEPENDE DEL PROBLEMA

COMPONENTES DEL ALGORITMO

# Representación

---

Debemos disponer de un mecanismo para codificar un individuo como un genotipo.

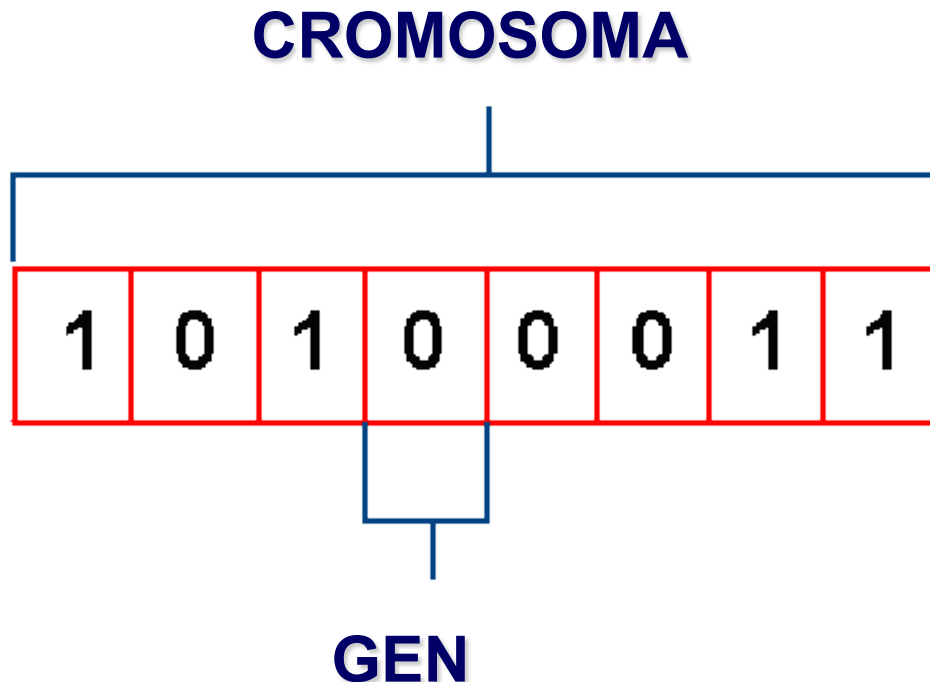
Existen muchas maneras de hacer esto y se ha de elegir la más relevante para el problema en cuestión.

Una vez elegida una representación, hemos de tener en mente como los genotipos (codificación) serán evaluados y qué operadores genéticos hay que utilizar.

# Ejemplo: Representación binaria

---

- La representación de un individuo se puede hacer mediante una codificación discreta, y en particular binaria.

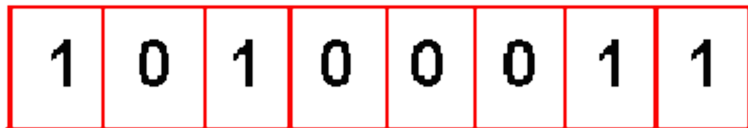




# Ejemplo: Representación binaria

---

**8 bits Genotipo**



**Fenotipo**

- **Entero**
- **Número real**
- **secuencia**
- ...
- **Cualquier otra?**

# Ejemplo: Representación binaria

---

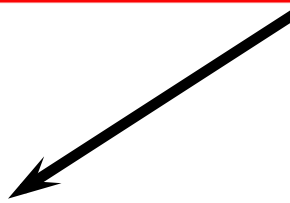
El fenotipo pueden ser números enteros

**Genotipo:**

1	0	1	0	0	0	1	1
---	---	---	---	---	---	---	---

**Fenotipo:**

**= 163**


$$1 \cdot 2^7 + 0 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 =$$
$$128 + 32 + 2 + 1 = 163$$

# Ejemplo: Representación binaria

---

El fenotipo pueden ser números reales


Ejemplo: un número entre 2.5 y 20.5 utilizando 8 dígitos binarios

**Genotipo:**

1	0	1	0	0	0	1	1
---	---	---	---	---	---	---	---

**Fenotipo:**

**= 13.9609**


$$x = 2.5 + \frac{163}{256} (20.5 - 2.5) = 13.9609$$

# Ejemplo: Representación Real

---

- Una forma natural de codificar una solución es utilizando valores reales como genes
- Muchas aplicaciones tienen esta forma natural de codificación

# Ejemplo: Representación Real

---

- Los individuos se representan como vectores de valores reales:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, x_i \in R$$

- La función de evaluación asocia a un vector un valor real de evaluación:

$$f : R^n \rightarrow R$$

# Ejemplo: Representación de orden

---

- Los individuos se representan como permutaciones.

7	3	6	8	2	4	1	5
---	---	---	---	---	---	---	---

- Se utilizan para problemas de secuenciación.
- Ejemplo famoso: Viajante de Comercio, donde cada ciudad tiene asignado un único número entre 1 y  $n$ .
- Necesita operadores especiales para garantizar que el resultado de aplicar un operador sigue siendo una permutación.

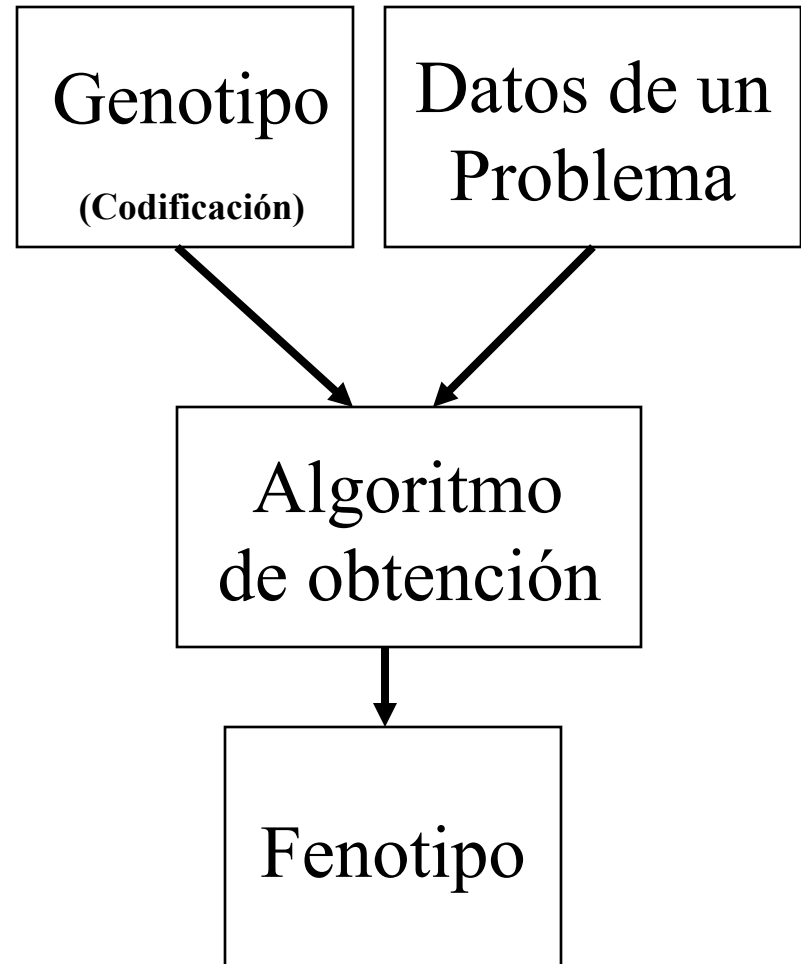
# Inicialización

---

- Uniforme sobre el espacio de búsqueda ... (si es posible)
  - Cadena binaria: 0 ó 1 con probabilidad 0.5
  - Representación real: uniforme sobre un intervalo dado (para valores acotados)
- Elegir la población a partir de los resultados de una heurística previa.

# Correspondencia entre Genotipo y Fenotipo

- Algunas veces la obtención del fenotipo a partir del genotipo es un proceso obvio.
- En otras ocasiones el genotipo puede ser un conjunto de parámetros para algún algoritmo, el cual trabaja sobre los datos de un problema para obtener un fenotipo





# Evaluación de un individuo

---

- Este es el paso más **costoso** para una aplicación real
- Puede ser una subrutina, un simulador, o cualquier proceso externo (ej. Experimentos en un robot, ....)
- Se pueden utilizar funciones aproximadas para reducir el costo de evaluación.
- Cuando hay restricciones, éstas se pueden introducir en el costo como penalización.
- Con múltiples objetivos se busca una solución de compromiso.

# ¿CÓMO SE CONSTRUYE UN AG?

---



# Estrategia de Selección

---

Debemos de garantizar que los mejores individuos tienen una mayor posibilidad de ser padres (reproducirse) frente a los individuos menos buenos.

Debemos de ser cuidadosos para dar una oportunidad de reproducirse a los individuos menos buenos. Éstos pueden incluir material genético útil en el proceso de reproducción.

Esta idea nos define la **presión selectiva** que determina en qué grado la reproducción está dirigida por los mejores individuos.

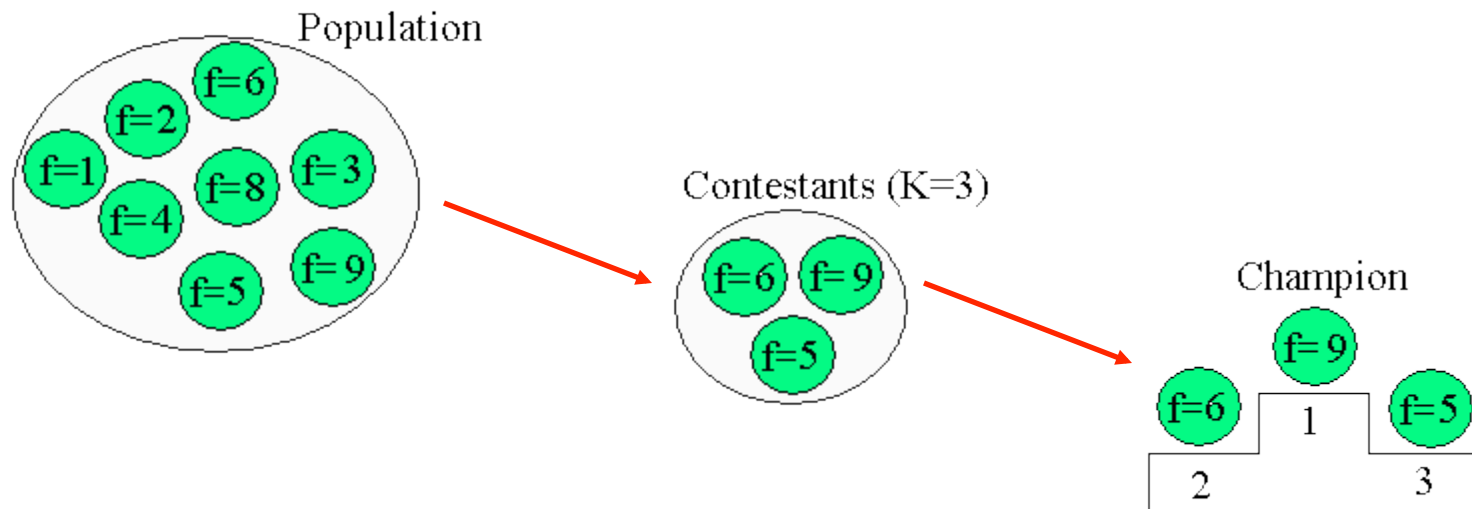
# Estrategia de Selección

## Selección por torneo

Para cada padre a seleccionar:

- Escoger aleatoriamente  $k$  individuos, con reemplazamiento
- Seleccionar el mejor de ellos

$k$  se denomina **tamaño del torneo**. A mayor  $k$ , mayor presión selectiva y viceversa.



# Estrategia de Selección

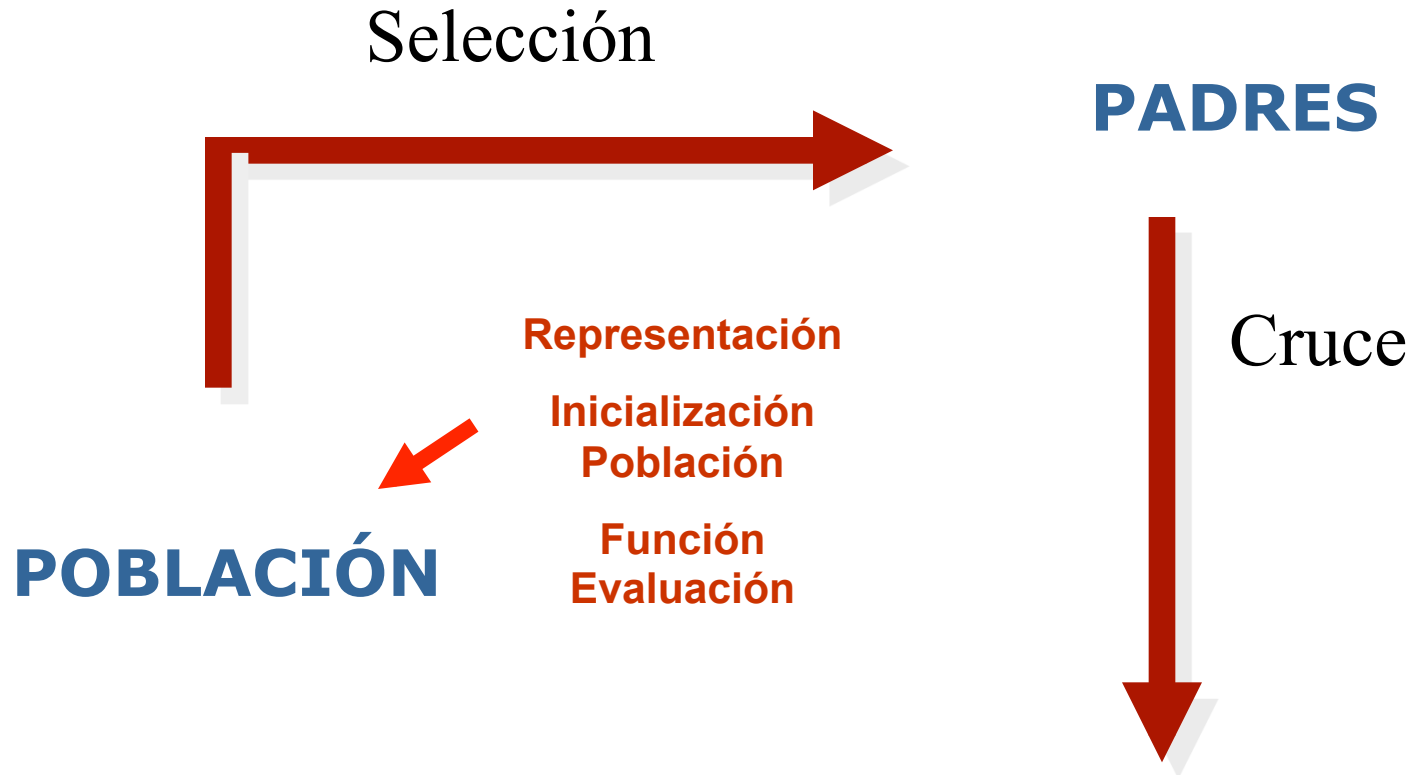
## Algunos esquemas de selección

---

- **Selección por Torneo (TS):** Escoge al individuo de mejor fitness de entre  $N_{ts}$  individuos seleccionados aleatoriamente ( $N_{ts}=2,3, \dots$ ).
- **Orden Lineal (LR):** La población se ordena en función de su fitness y se asocia una probabilidad de selección a cada individuo que depende de su orden.
- **Selección Aleatoria (RS).**
- **Emparejamiento Variado Inverso (NAM):** Un padre lo escoge aleatoriamente, para el otro selecciona  $N_{nam}$  padres y escoge el más lejano al primer ( $N_{nam}=3,5, \dots$ ). Está orientado a generar diversidad.
- **Selección por Ruleta:** Se asigna una probabilidad de selección proporcional al valor del fitness del cromosoma. (Modelo clásico)

# ¿CÓMO SE CONSTRUYE UN AG?

---



# Operador de Cruce

---

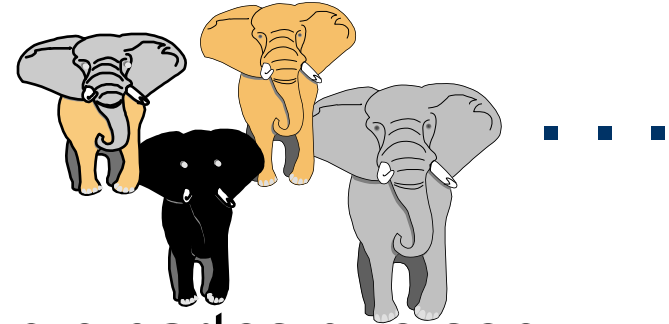
Podríamos tener uno o más operadores de cruce para nuestra representación.

Algunos aspectos importantes a tener en cuenta son:

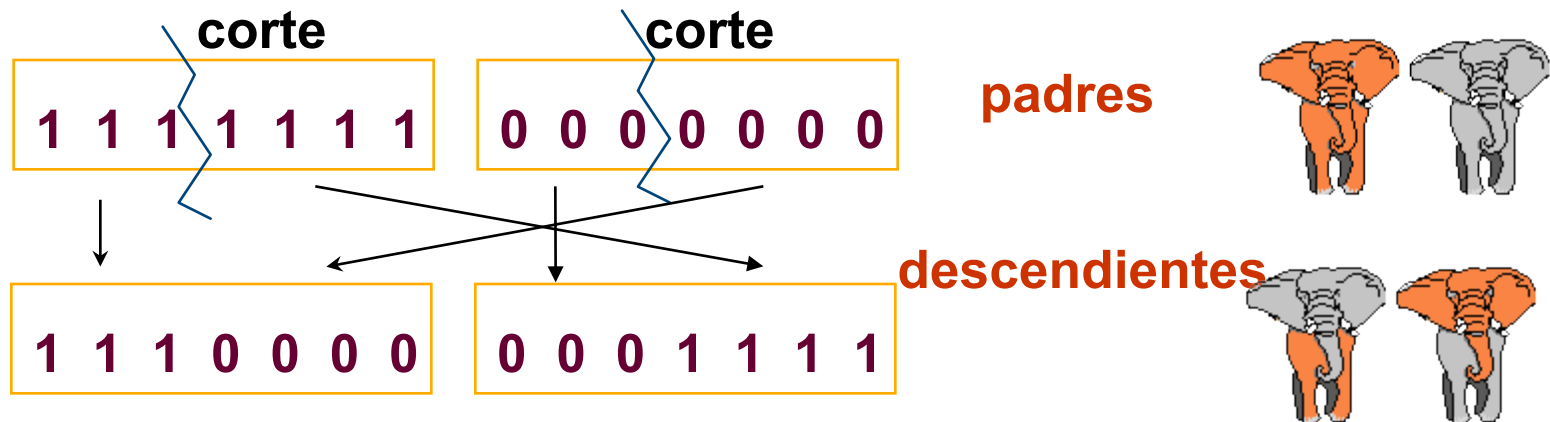
- Los hijos deberían heredar algunas características de **cada** padre. Si éste no es el caso, entonces estamos ante un operador de mutación.
- Se debe diseñar de acuerdo a la representación.
- La recombinación debe producir cromosomas válidos.
- Se utiliza con una probabilidad alta de actuación sobre cada pareja de padres a cruzar ( $P_c$  entre 0.6 y 0.9), si no actúa los padres son los descendientes del proceso de recombinación de la pareja.

# Ejemplo: Operador de cruce para representación binaria

Población:



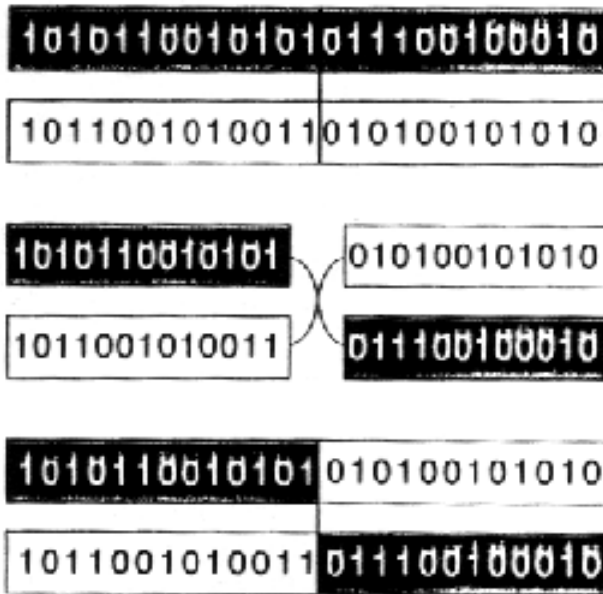
Cada cromosoma se corta en  $n$  partes que son re combinadas. (Ejemplo para  $n = 1$ ).





# Operador de Cruce

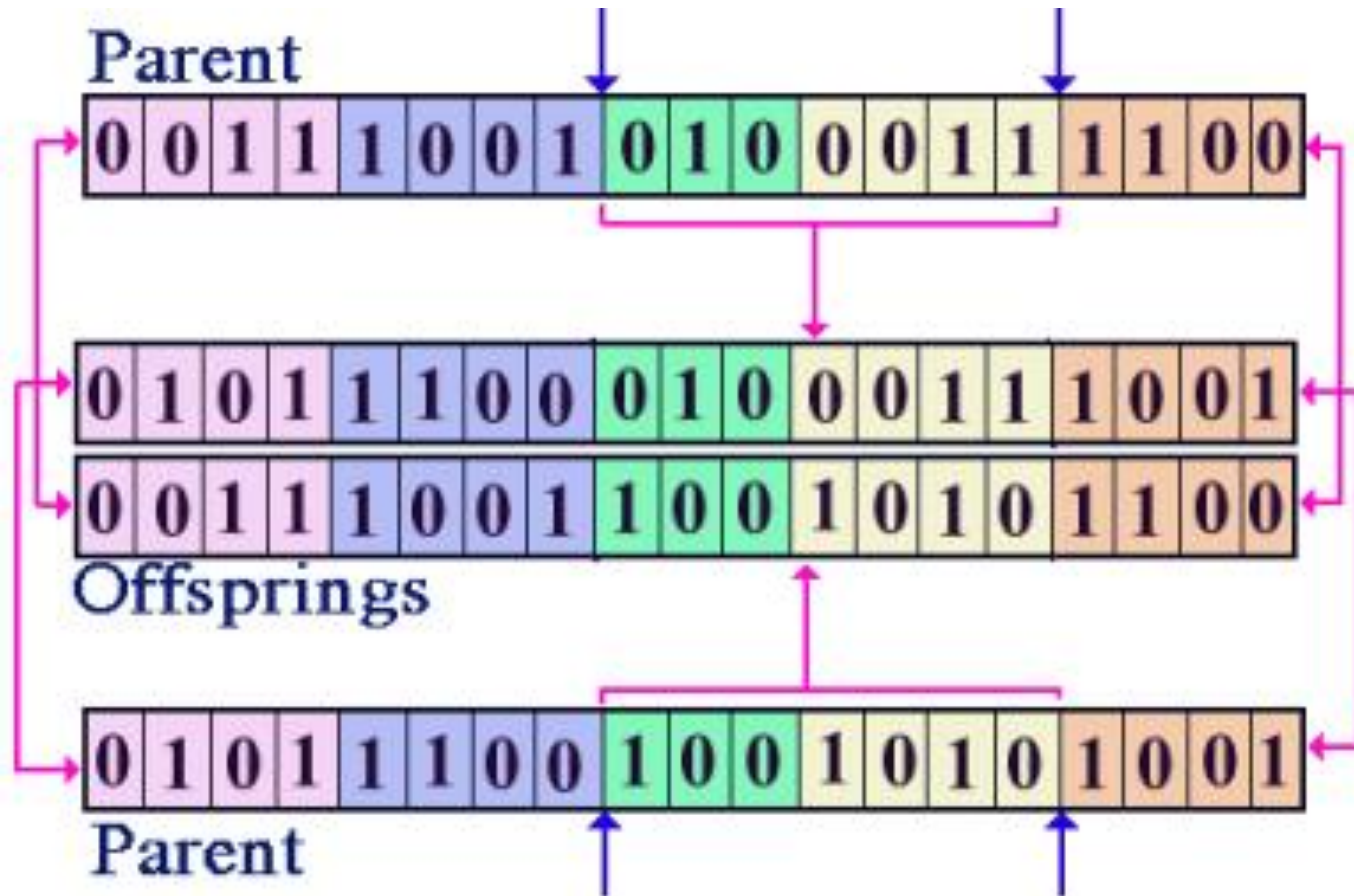
**Imagen clásica (John Holland) que introduce el operador de cruce**



CROSSOVER is the fundamental mechanism of genetic rearrangement for both real organisms and genetic algorithms.

Chromosomes line up and then swap the portions of their genetic code beyond the crossover point.

# Ejemplo: Operador de cruce en dos puntos para representación binaria



# Ejemplo: Operador de cruce para representación real

---

Recombinación discreta (cruce uniforme): dados 2 padres se crea un descendiente como sigue:

a	b	c	d	e	f	g	h
---	---	---	---	---	---	---	---



a	b	C	d	E	f	g	H
---	---	---	---	---	---	---	---

A	B	C	D	E	F	G	H
---	---	---	---	---	---	---	---

Existe muchos operadores específicos para la codificación real.

# Ejemplo: Operador de cruce para representación real

---

Recombinación aritmética (cruce aritmético):

a	b	c	d	e	f
---	---	---	---	---	---

A	B	C	D	E	F
---	---	---	---	---	---



$(a+A)/2$	$(b+B)/2$	$(c+C)/2$	$(d+D)/2$	$(e+E)/2$	$(f+F)/2$
-----------	-----------	-----------	-----------	-----------	-----------

# Ejemplo: Operador de cruce para representación real: **BLX- $\alpha$**

---

- Dados 2 cromosomas

$$C_1 = (c_{11}, \dots, c_{1n}) \text{ y } C_2 = (c_{21}, \dots, c_{2n}) ,$$

- BLX-  $\alpha$  genera dos descendientes

$$H_k = (h_{k1}, \dots, h_{ki}, \dots, h_{kn}) , k = 1, 2$$

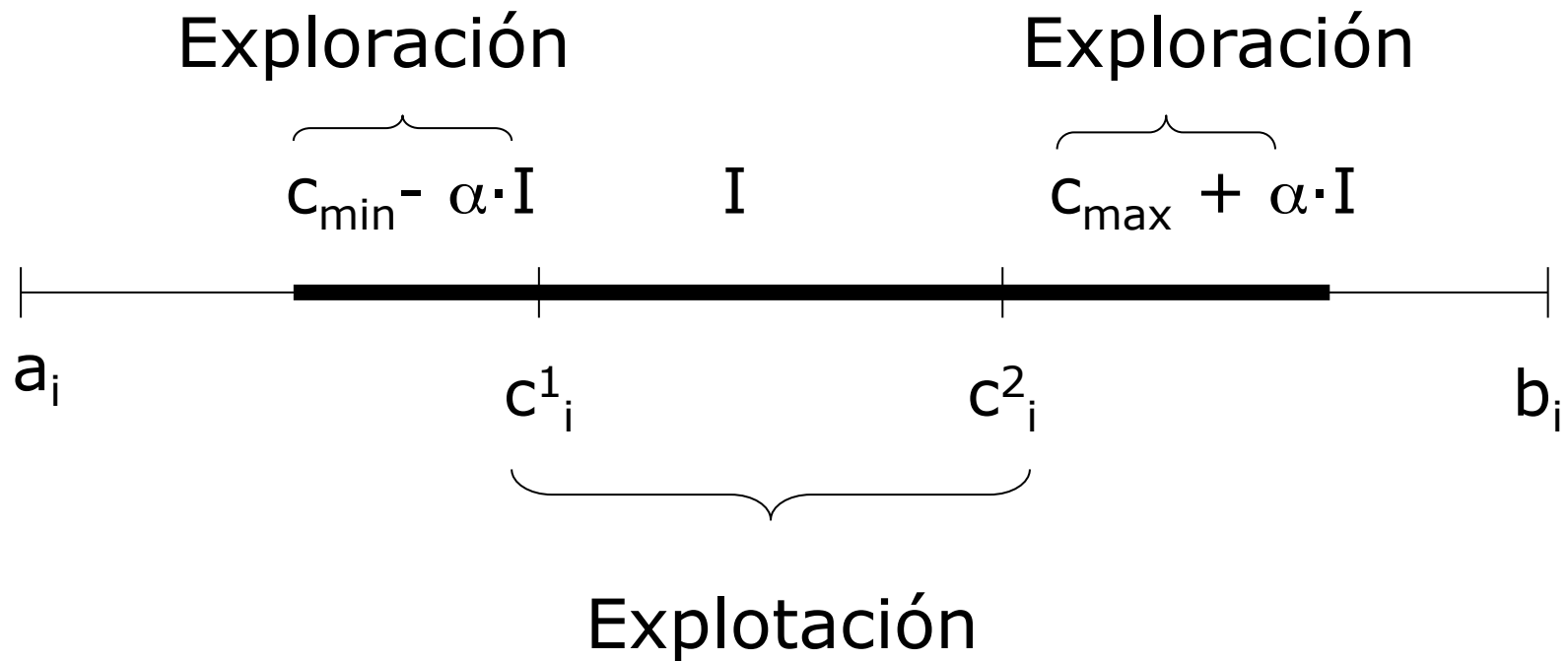
- donde  $h_{ki}$  se genera aleatoriamente en el intervalo:

$$[C_{\min} - I \cdot \alpha, C_{\max} + I \cdot \alpha]$$

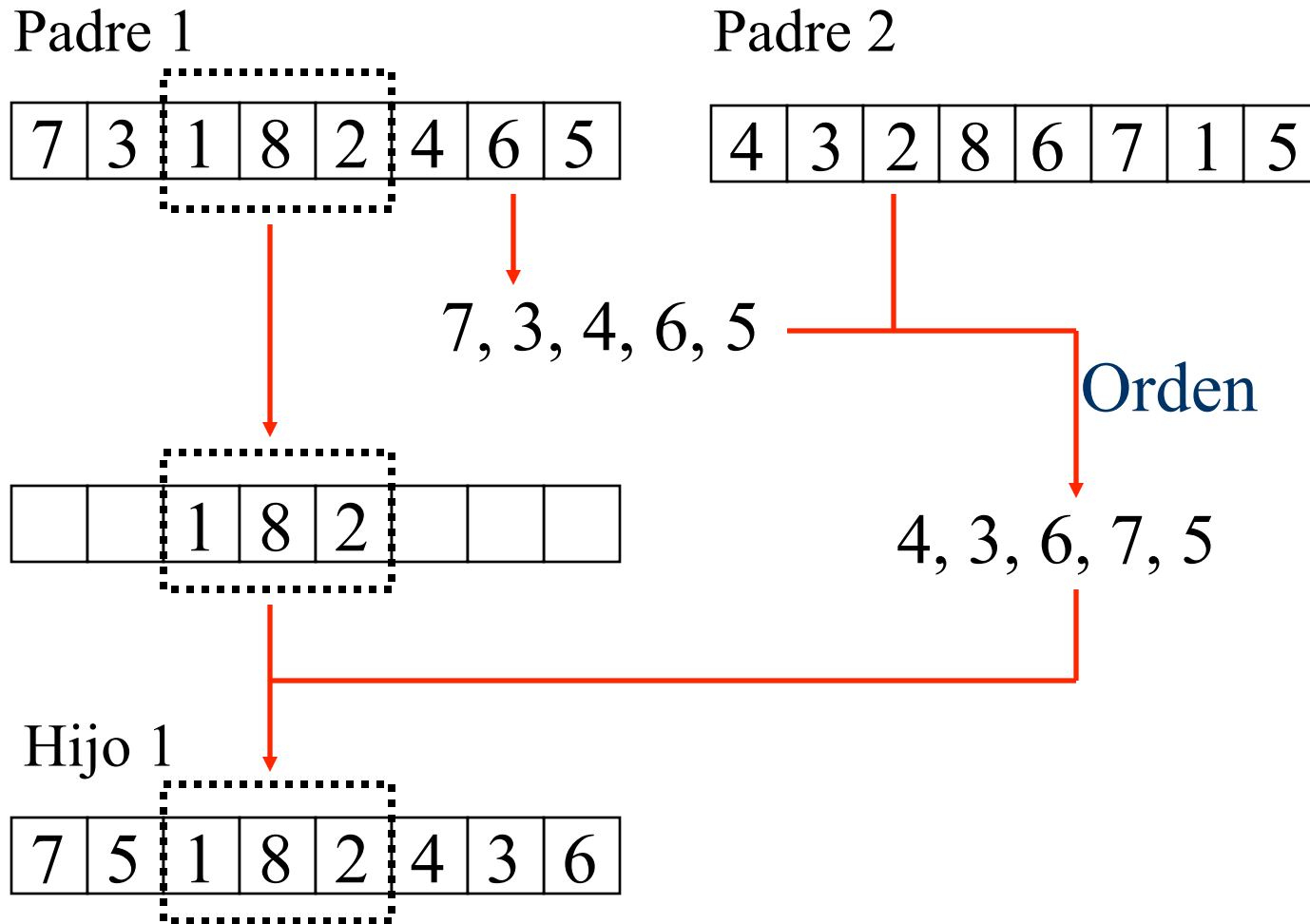
- $C_{\max} = \max \{c_{1i}, c_{2i}\}$
- $C_{\min} = \min \{c_{1i}, c_{2i}\}$
- $I = C_{\max} - C_{\min} , \alpha \in [0, 1]$

# Ejemplo: Operador de cruce para representación real: **BLX- $\alpha$**

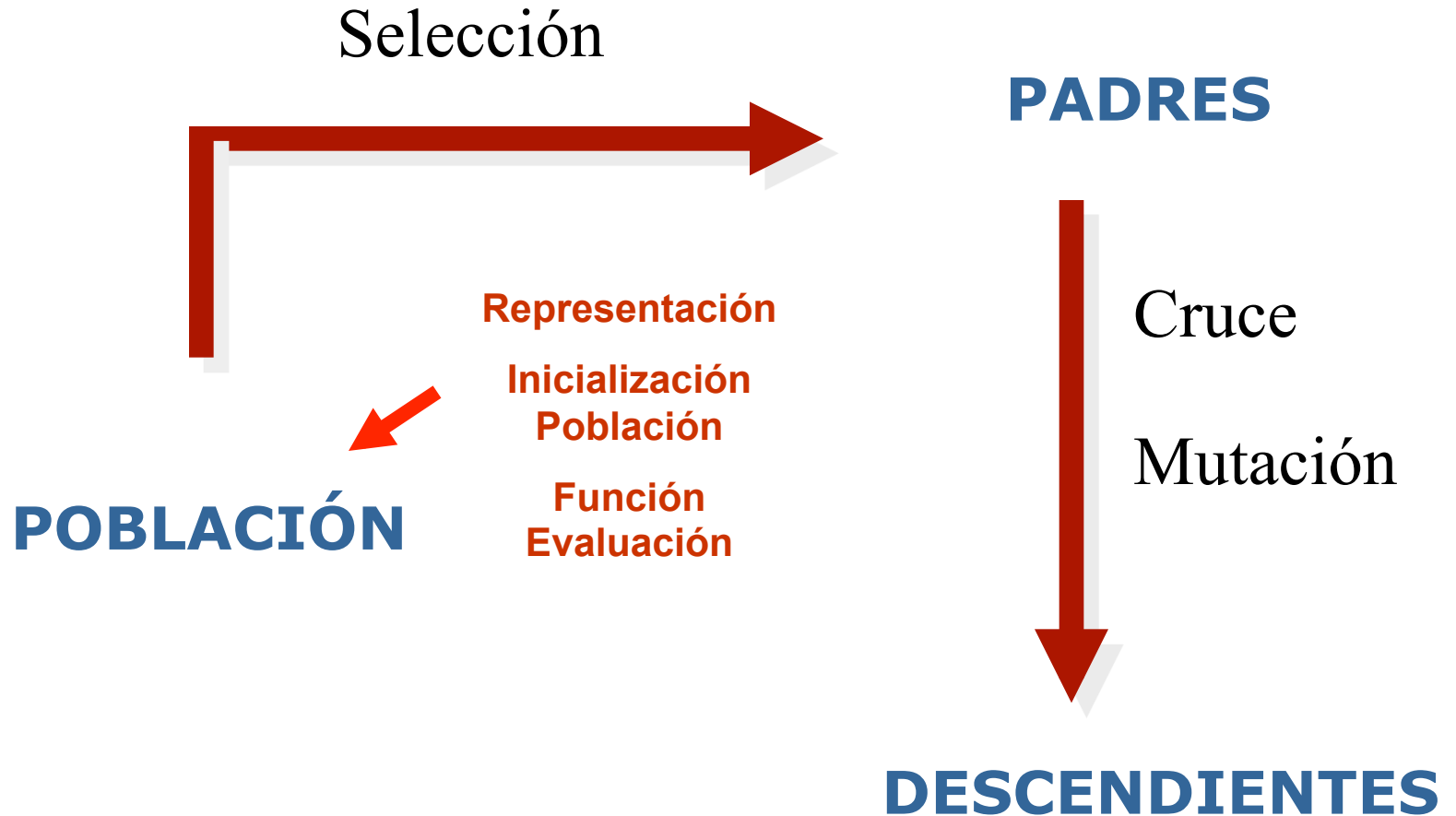
---



# Ejemplo: Operador de cruce para representación de orden OX



# ¿CÓMO SE CONSTRUYE UN AG?





# Operador de mutación

---

Podemos tener uno o más operadores de mutación para nuestra representación.

Algunos aspectos importantes a tener en cuenta son:

- Debe permitir alcanzar cualquier parte del espacio de búsqueda.
- El tamaño de la mutación debe ser controlado.
- Debe producir cromosomas válidos.
- Se aplica con una probabilidad muy baja de actuación sobre cada descendiente obtenido tras aplicar el operador de cruce (incluidos los descendientes que coinciden con los padres porque el operador de cruce no actúa).

# Ejemplo: Mutación para representación discreta binaria

---

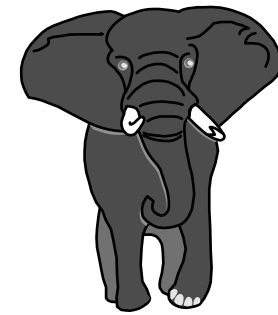
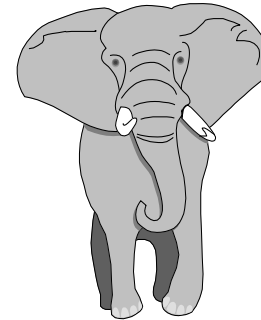
antes

1 1 1 1 1 1 1

después

1 1 1 0 1 1 1

↑  
gen mutado



La mutación ocurre con una probabilidad  $p_m$  para cada gen

# Ejemplo: Mutación para representación real

---

Perturbación de los valores mediante un valor aleatorio.

Frecuentemente, mediante una distribución Gaussiana/normal  $N(0, \sigma)$ , donde

- 0 es la media
- $\sigma$  es la desviación típica

$$x'_i = x_i + N(0, \sigma_i)$$

para cada parámetro.

# Ejemplo: Mutación para representación de orden

---

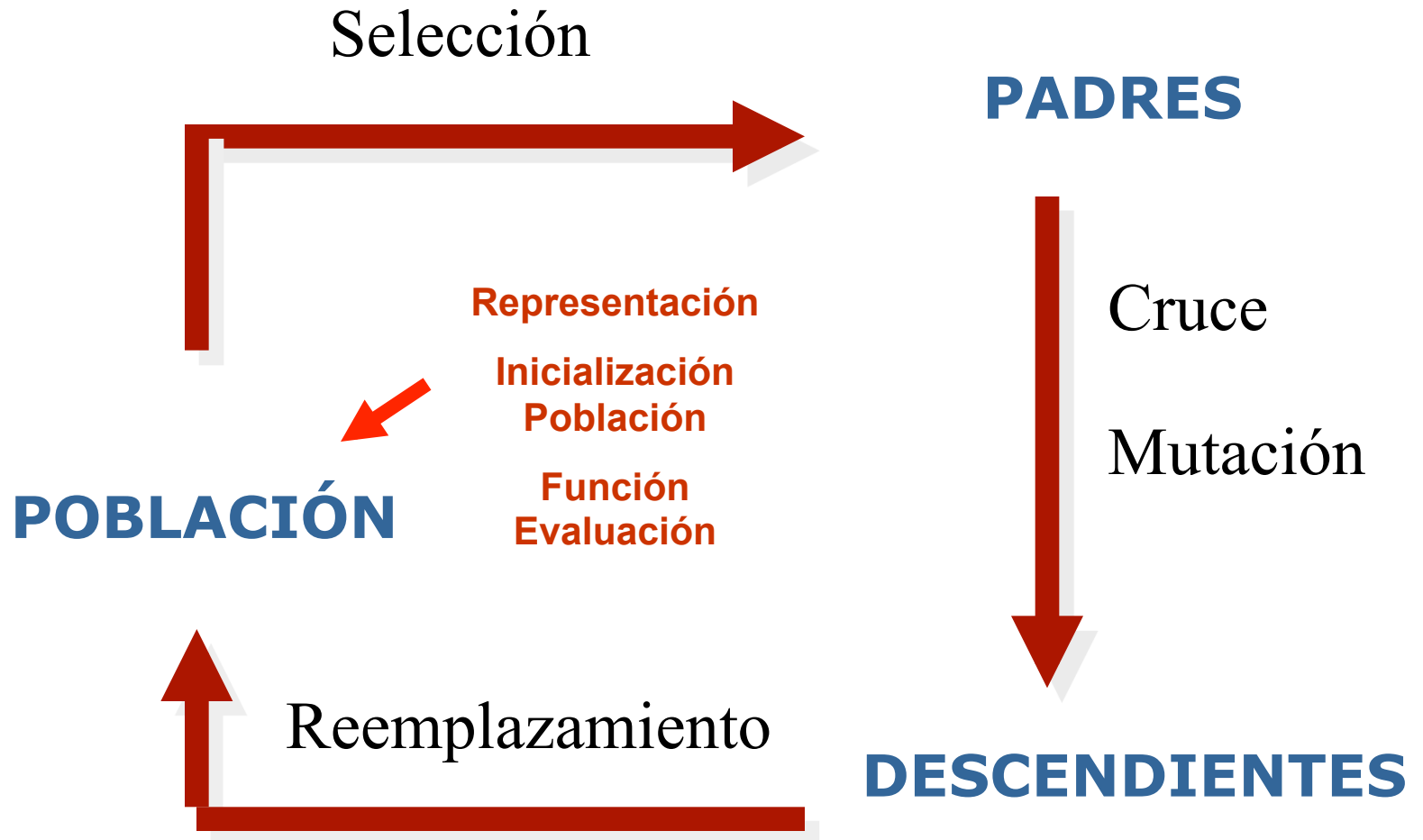
Selección aleatoria de dos genes e intercambio de ambos.

7	3	1	8	2	4	6	5
---	---	---	---	---	---	---	---



7	3	6	8	2	4	1	5
---	---	---	---	---	---	---	---

# ¿CÓMO SE CONSTRUYE UN AG?



# Estrategia de Reemplazamiento

---

La estrategia de reemplazamiento determina qué individuos son reemplazados de la población y los que entran en ella.

En el modelo generacional, donde se genera una nueva población de descendientes, ésta sustituye completamente a la anterior.

Podemos decidir no reemplazar al mejor cromosoma de la población: **Elitismo**

**(el uso del Elitismo es aconsejado en los modelos generacionales para no perder la mejor solución encontrada).**

# Criterio de Parada

---

- Cuando se alcanza el óptimo!
- Recursos limitados de CPU:  
Fijar el máximo número de evaluaciones
- Límite sobre la paciencia del usuario: Después de algunas iteraciones sin mejora.

# ¿CÓMO SE CONSTRUYE UN AG?

## RESUMEN





# 3. SOBRE SU UTILIZACIÓN

---

- **Nunca** sacar conclusiones de una única ejecución
  - utilizar medidas estadísticas (medias, medianas, ...)
  - con un número suficiente de ejecuciones independientes
- “Se puede obtener lo que se desea en una experimentación de acuerdo a la dificultad de los casos utilizados” – No se debe ajustar/chequear la actuación de un algoritmo sobre ejemplos simples si se desea trabajar con casos reales
- Desde el punto de vista de las aplicaciones:  
doble enfoque y diferente diseño
  - Encontrar una solución **muy buena** al menos una vez
  - Encontrar al menos una solución **muy buena** en cada ejecución

# 4. ALGUNAS CUESTIONES: DIVERSIDAD, EXPLORACIÓN vs EXPLOTACIÓN

---

## Diversidad genética

- Asociada a las diferencias entre los cromosomas en la población
- Falta de diversidad genética = todos los individuos en la población son parecidos
- *Falta de diversidad* → *convergencia al vecino más cercano*
- *Alta presión selectiva* → *falta de diversidad*
- En la práctica es irreversible. **Soluciones:**
  - *Inclusión de mecanismos de diversidad en la evolución*
  - *Reinicialización cuando se produce convergencia prematura*

# ALGUNAS CUESTIONES: DIVERSIDAD, EXPLORACIÓN vs EXPLOTACIÓN

---

## Exploración vs Explotación

- **Exploración** = muestrear regiones desconocidas

*Excesiva exploración = búsqueda aleatoria, no convergencia*

- **Explotación** = trata de mejorar el mejor individuo

*Excesiva explotación = solo búsqueda local ... convergencia a un óptimo local*

## 5. MODELOS: GENERACIONAL vs ESTACIONARIO

---

**Modelo generacional:** Durante cada iteración se crea una población completa con nuevos individuos

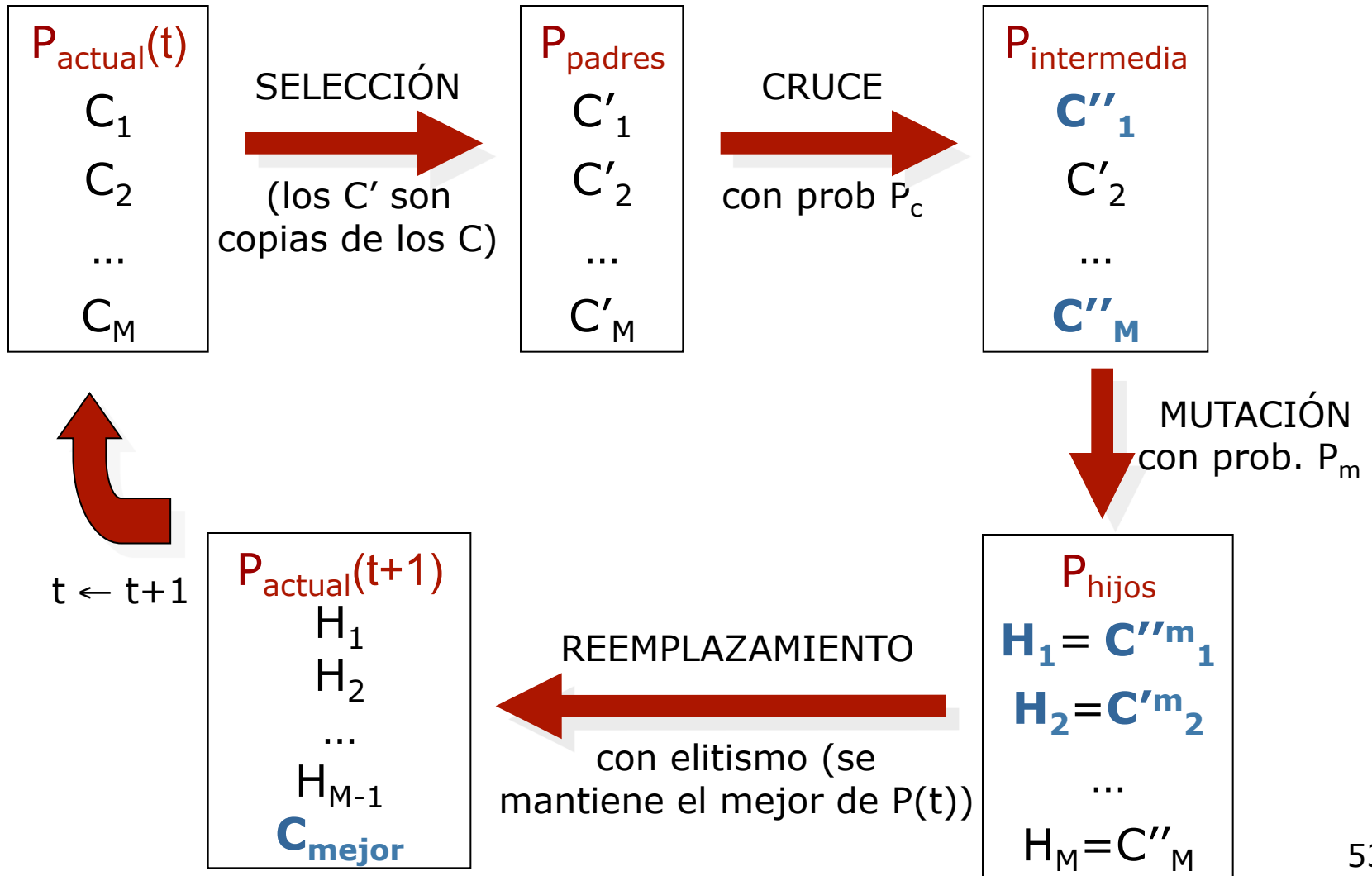
**La nueva población reemplaza directamente a la antigua**

**Modelo estacionario:** Durante cada iteración se escogen dos padres de la población (diferentes mecanismos de muestreo) y se les aplican los operadores genéticos

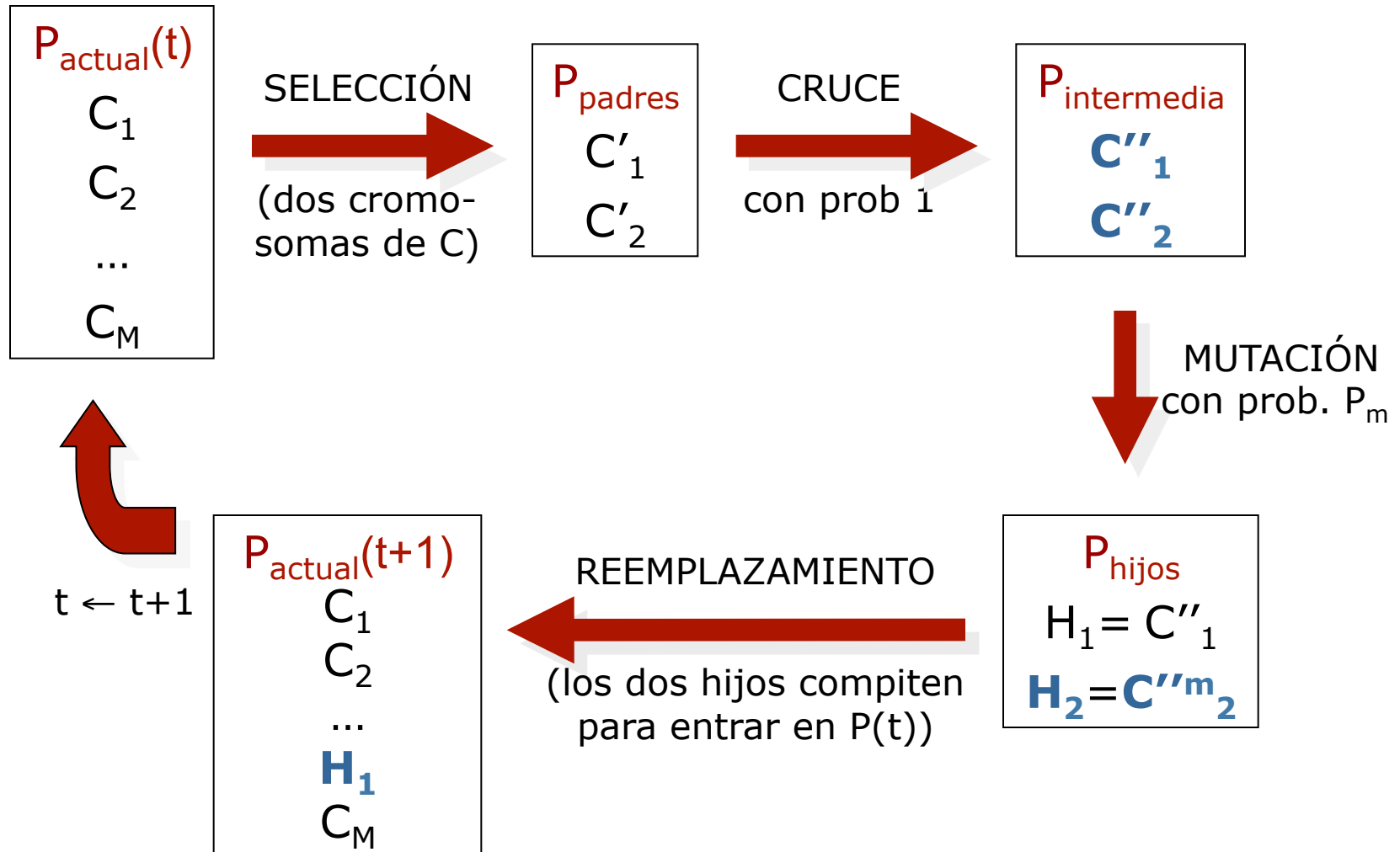
**El/los descendiente/s reemplaza/n a uno/dos cromosoma/s de la población inicial**

*El modelo estacionario produce una presión selectiva alta (convergencia rápida) cuando se reemplazan los peores cromosomas de la población*

# Modelo Generacional



# Modelo Estacionario



# Estrategia de Reemplazamiento

## Algunas estrategias de reemplazo para AG estacionarios

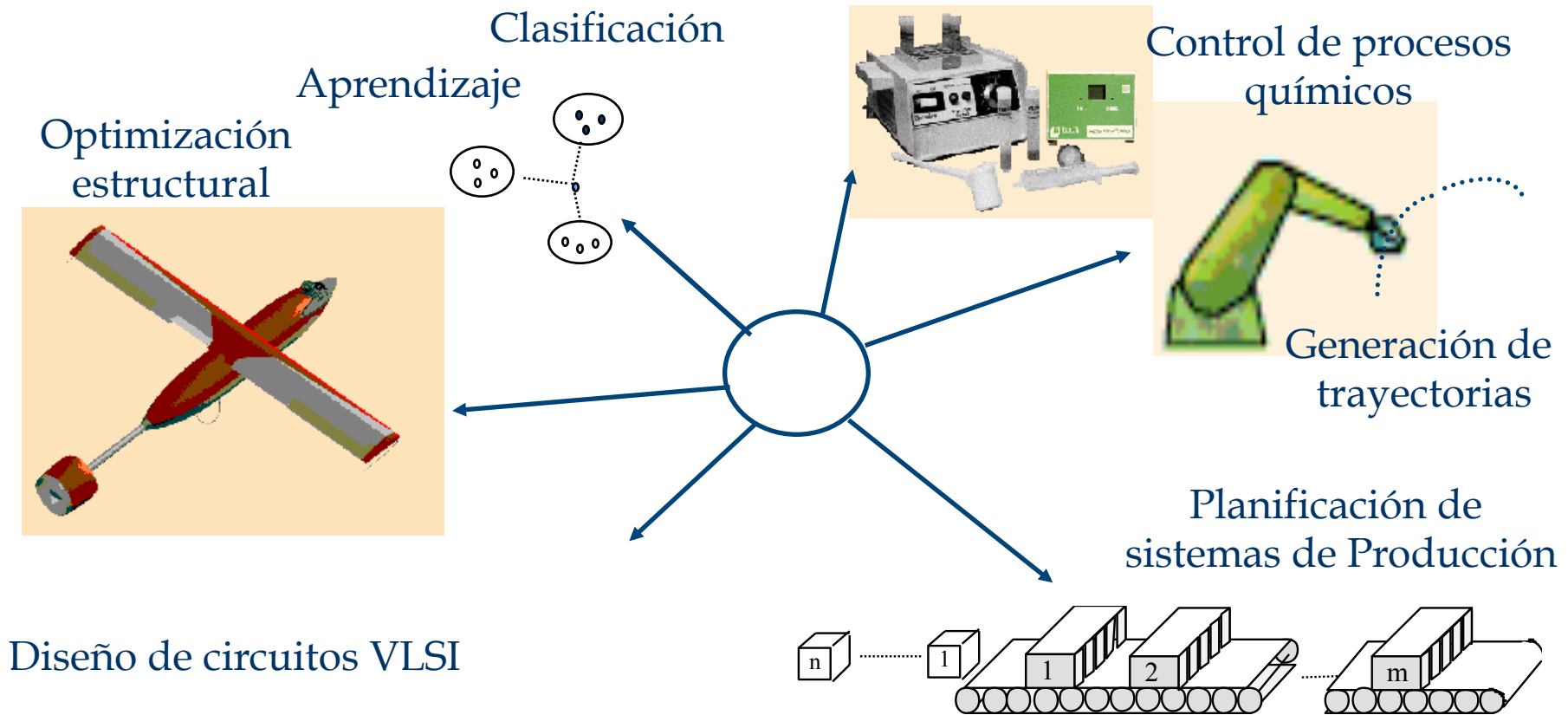
---

Cuando se considera un modelo estacionario (en el que se reemplazan solo uno o dos padres, frente al modelo generacional en el que se reemplaza la población completa), nos encontramos con diferentes propuestas.

A continuación presentamos algunas posibilidades:

- **Reemplazar al peor de la población (RW)**. Genera alta presión selectiva.
- **Torneo Restringido (RTS)**: Se reemplaza al más parecido de entre  $w$  ( $w=3, \dots$ ). Mantiene una cierta diversidad.
- **Peor entre semejantes (WAMS)**: Se reemplaza el peor cromosoma del conjunto de los  $w$  ( $w=3, \dots$ ) padres más parecidos al descendiente generado (seleccionados de toda la población). Busca equilibrio entre diversidad y presión selectiva.
- **Algoritmo de Crowding Determinístico (DC)**: El hijo reemplaza a su padre más parecido. Mantiene diversidad.

# 6. DOMINIOS DE APLICACIÓN



Diseño de circuitos VLSI



# DOMINIOS DE APLICACIÓN

---

- **Optimización combinatoria y en dominios reales**
- **Modelado e identificación de sistemas**
- **Planificación y control**
- **Ingeniería**
- **Vida artificial**
- **Aprendizaje y minería de datos**
- **Internet y Sistemas de Recuperación de Información**
- **...**

**Ref: T. Bäck, D.B. Fogel, Z. Michalewicz, Handbook of Evolutionary Computation. Oxford Univ. Press, 1997.**

# 7. EJEMPLO: VIAJANTE DE COMERCIO

---

## Representación de orden

**(3 5 1 13 6 15 8 2 17 11 14 4 7 9 10 12 16)**

17 ciudades

Objetivo: Suma de la distancia entre las ciudades.

Población: 61 cromosomas - Elitismo

Cruce: OX ( $P_c = 0,6$ )

Mutación: Inversión de una lista ( $P_m = 0,01$  - cromosoma)

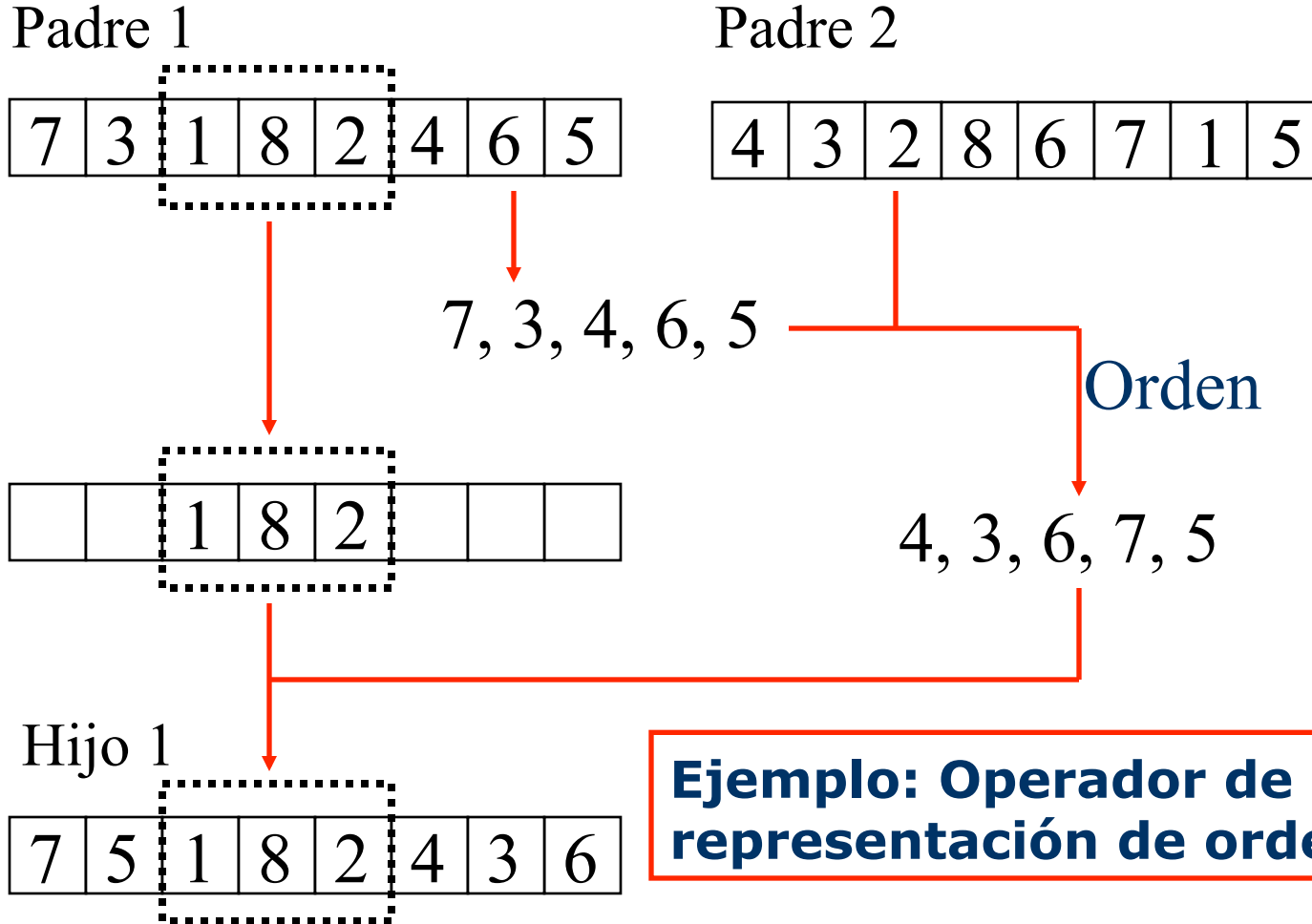
# 7. EJEMPLO: VIAJANTE DE COMERCIO

---

- 1. Generación de la solución inicial:** aleatoria o greedy
- 2. Esquema de representación:** Representación de orden mediante permutación  $\{1, \dots, n\}$
- 3. Selección:** torneo binario
- 4. Enfoque:** generacional con elitismo (1 individuo)
- 5. Operador de cruce:** Operador OX.
- 6. Operador de mutación:** intercambio de 2 genes
- 7. Función objetivo (minimización):** La distancia recorrida por el viajante.

$$C(S) = \sum_{i=1}^{n-1} (D[S[i], S[i+1]]) + D[S[n], S[1]]$$

# 7. EJEMPLO: VIAJANTE DE COMERCIO



**Ejemplo: Operador de cruce para representación de orden OX**

# 7. EJEMPLO: VIAJANTE DE COMERCIO

## Otros operadores de cruce para la representación de orden

### Cruce para representación de orden PMX

- Se elige una subcadena central y se establece una correspondencia por posición entre las asignaciones contenidas en ellas
- Cada hijo contiene la subcadena central de uno de los padres y el mayor número posible de asignaciones en las posiciones definidas por el otro padre. Cuando se forma un ciclo, se sigue la correspondencia fijada para incluir una asignación nueva

$$\text{Padre}_1 = (1 \ 2 \ 3 \ | \ 4 \ 5 \ 6 \ 7 \ | \ 8 \ 9)$$

$$\text{Padre}_2 = (4 \ 5 \ 3 \ | \ 1 \ 8 \ 7 \ 6 \ | \ 9 \ 2)$$

$$\text{Hijo}'_1 = (* \ * \ * \ | \ 1 \ 8 \ 7 \ 6 \ | \ * \ *)$$

$$\text{Hijo}'_2 = (* \ * \ * \ | \ 4 \ 5 \ 6 \ 7 \ | \ * \ *)$$

Correspondencias: (1-4, 8-5, 7-6, 6-7)

$$\text{Hijo}_1 = (1-4 \ 2 \ 3 \ | \ 1 \ 8 \ 7 \ 6 \ | \ 8-5 \ 9) = (4 \ 2 \ 3 \ | \ 1 \ 8 \ 7 \ 6 \ | \ 5 \ 9)$$

$$\text{Hijo}_2 = (4-1 \ 5-8 \ 3 \ | \ 4 \ 5 \ 6 \ 7 \ | \ 9 \ 2) = (1 \ 8 \ 3 \ | \ 4 \ 5 \ 6 \ 7 \ | \ 9 \ 2)$$

# 7. EJEMPLO: VIAJANTE DE COMERCIO

## Otros operadores de cruce para la representación de orden

### Cruce para representación de orden CX

- Partiendo de la asignación  $i$  del primer padre, CX toma la siguiente asignación  $j$  del segundo padre como aquella en la misma posición que  $i$ , y sitúa  $j$  en la misma posición que ocupa en el primer padre

$$\text{Padre}_1 = (1\ 2\ 3\ 4\ 5\ 6\ 7\ 8)$$

$$\text{Padre}_2 = (2\ 4\ 6\ 8\ 7\ 5\ 3\ 1)$$

- Aleatoriamente podemos escoger 1 ó 2 para la primera posición, supongamos que escogemos 1. Esto implica escoger 8 en la posición 8, lo cual supone escoger 4 en la posición 4 y, por tanto, 2 en la posición 2

$$(1\ *\ *\ *\ *\ *\ *\ *) \rightarrow (1\ *\ *\ *\ *\ *\ *\ 8) \rightarrow (1\ *\ *\ 4\ *\ *\ *\ 8) \rightarrow (1\ 2\ *\ 4\ *\ *\ *\ 8)$$

- Se ha formado un ciclo. Se escoge aleatoriamente una ciudad entre 3 ó 6 para la tercera posición, supongamos que escogemos 6

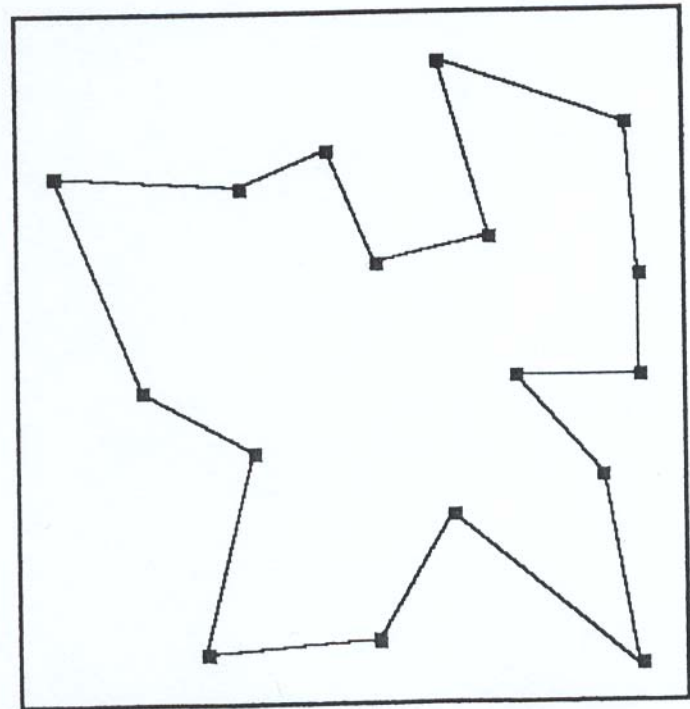
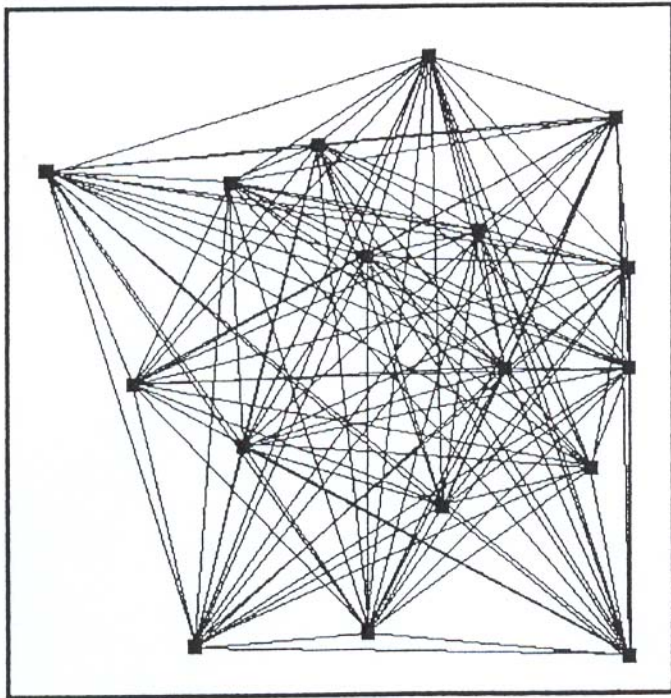
$$\text{Hijo} = (1\ 2\ 6\ 4\ *\ *\ *\ 8)$$

- Esto implica escoger obligatoriamente la ciudad 5 en la posición 6, lo que implica 7 en la posición 5 y 3 en la posición 7

$$\text{Hijo} = (1\ 2\ 6\ 4\ 7\ 5\ 3\ 8)$$

# Viajante de Comercio

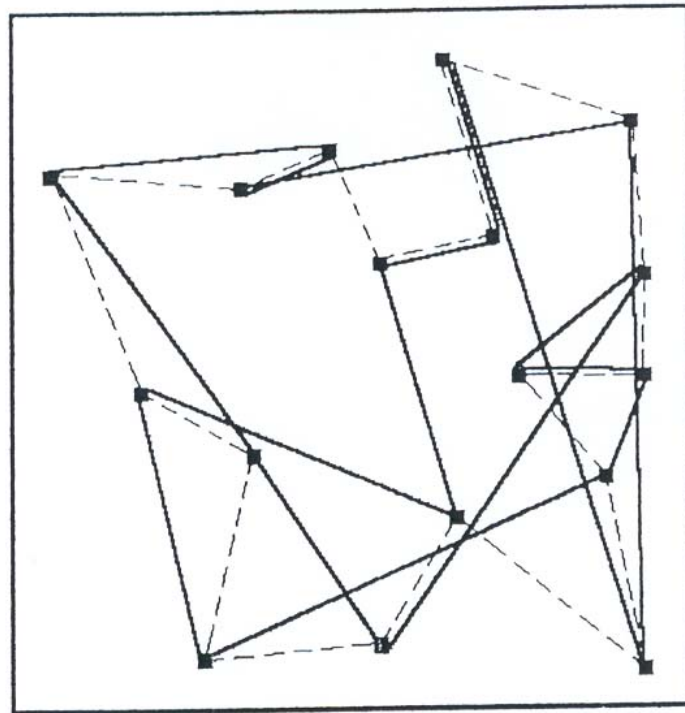
---



$17! = 3.5568743 \text{ e}14$  recorridos posibles

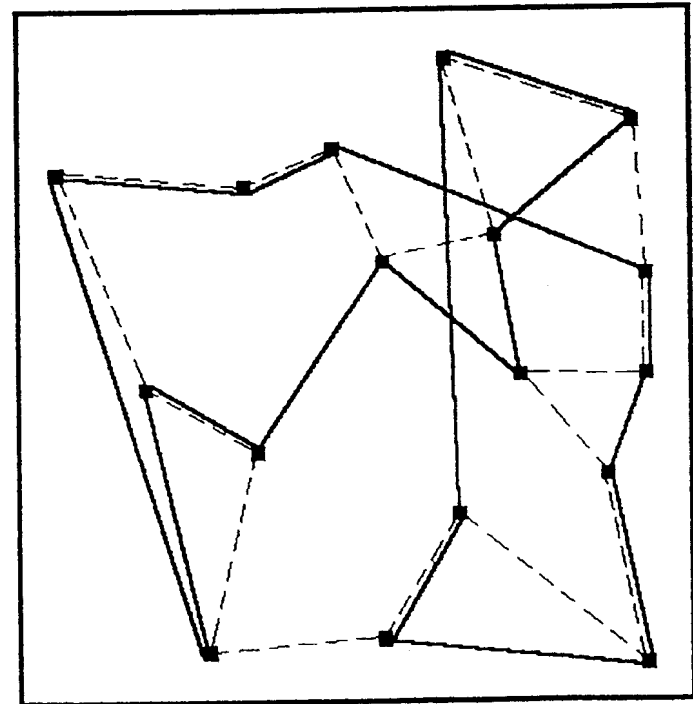
Solución óptima: 226.64

# Viajante de Comercio



— Mejor solución  
- - - Solución optimal

Iteración: 0 Costo: 403.7



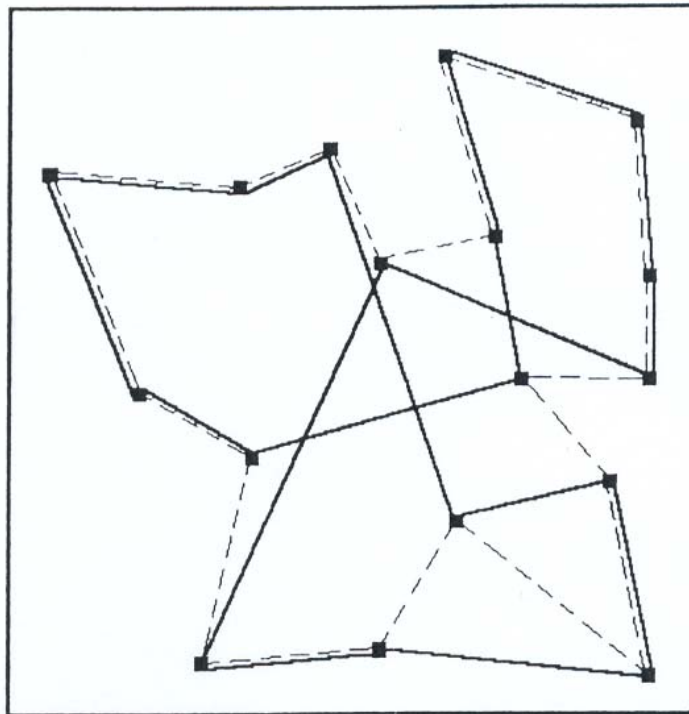
— Mejor solución  
- - - Solución optimal

Iteración: 25 Costo: 303.86

Solución óptima: 226.64

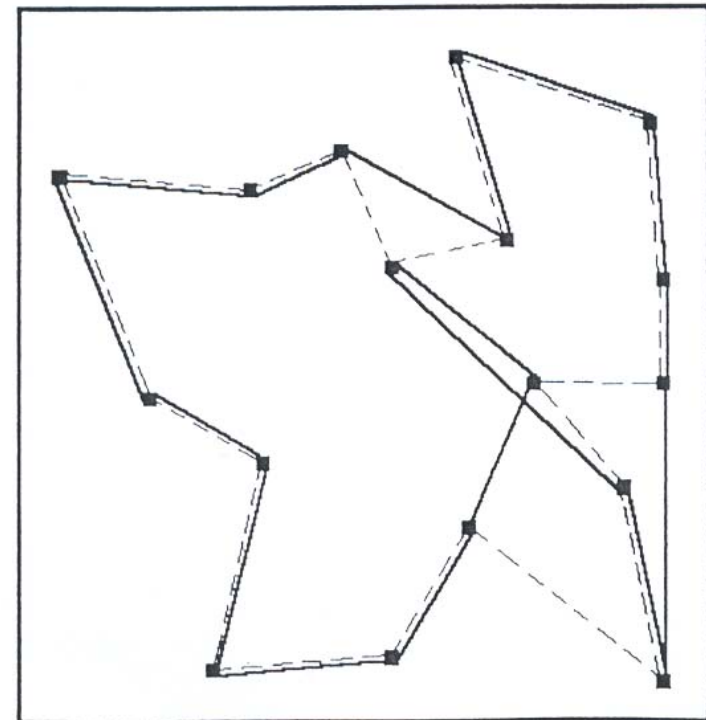


# Viajante de Comercio



—— Mejor solución  
- - - Solución optimal

Iteración: 50 Costo: 293,6

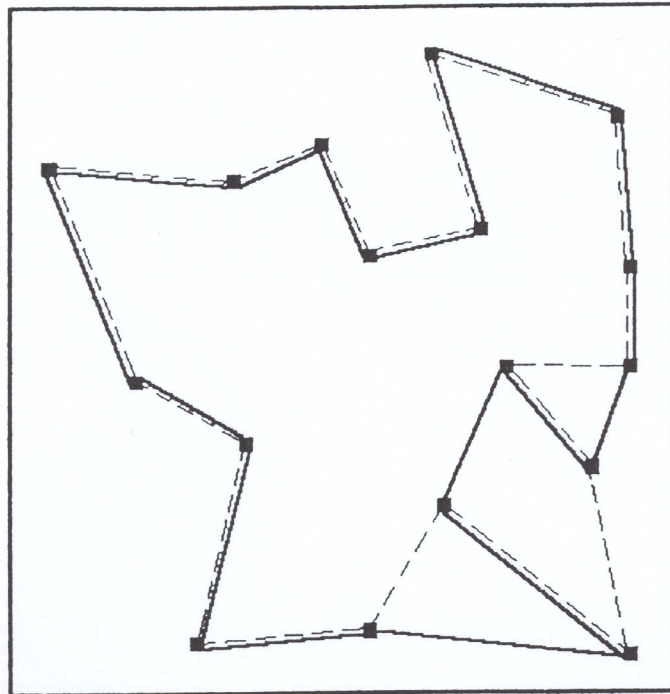


—— Mejor solución  
- - - Solución optimal

Iteración: 100 Costo: 256,55

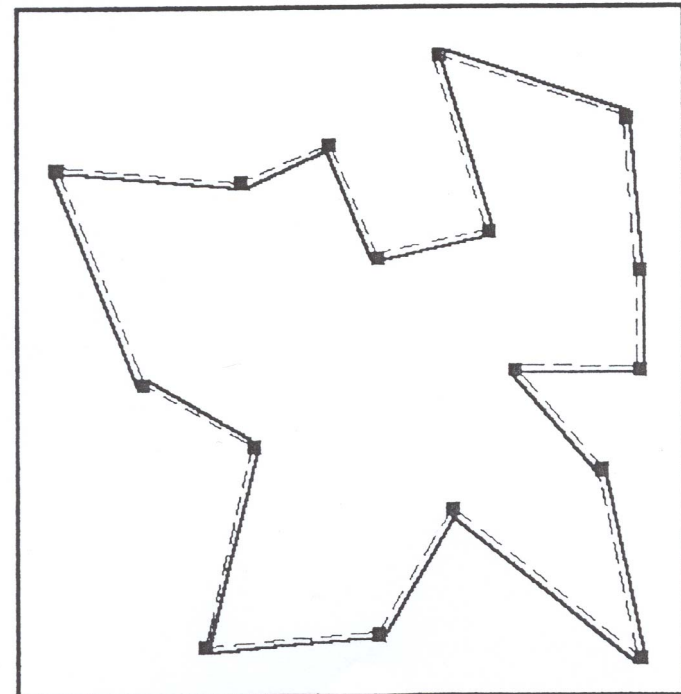
Solución óptima: 226,64

# Viajante de Comercio



— Mejor solución  
- - - Solución optimal

Iteración: 200 Costo: 231,4

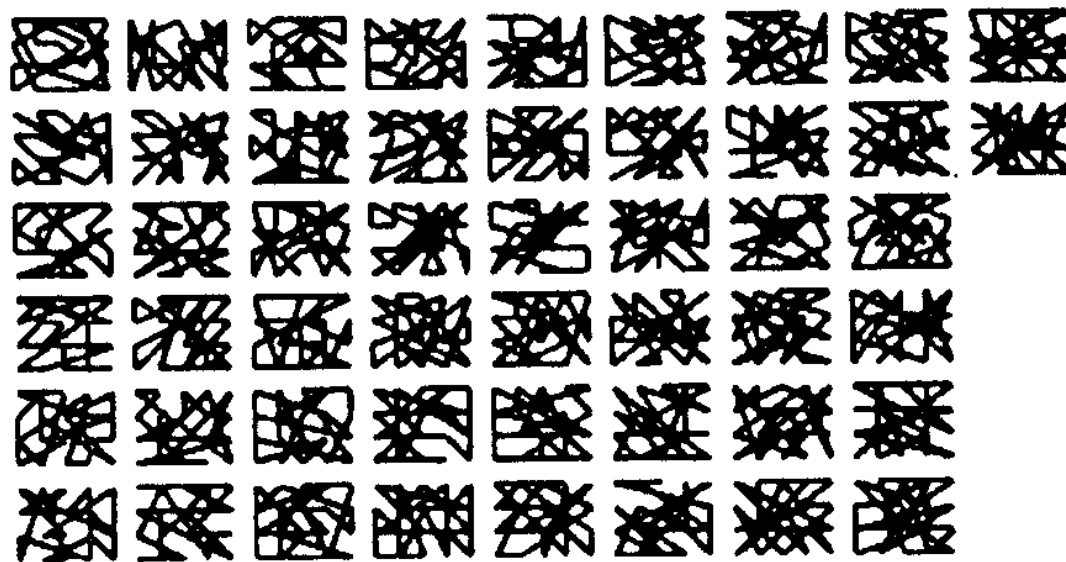


— Mejor solución  
- - - Solución optimal

Iteración: 250 Solución  
óptima: 226,64

# Viajante de Comercio

---

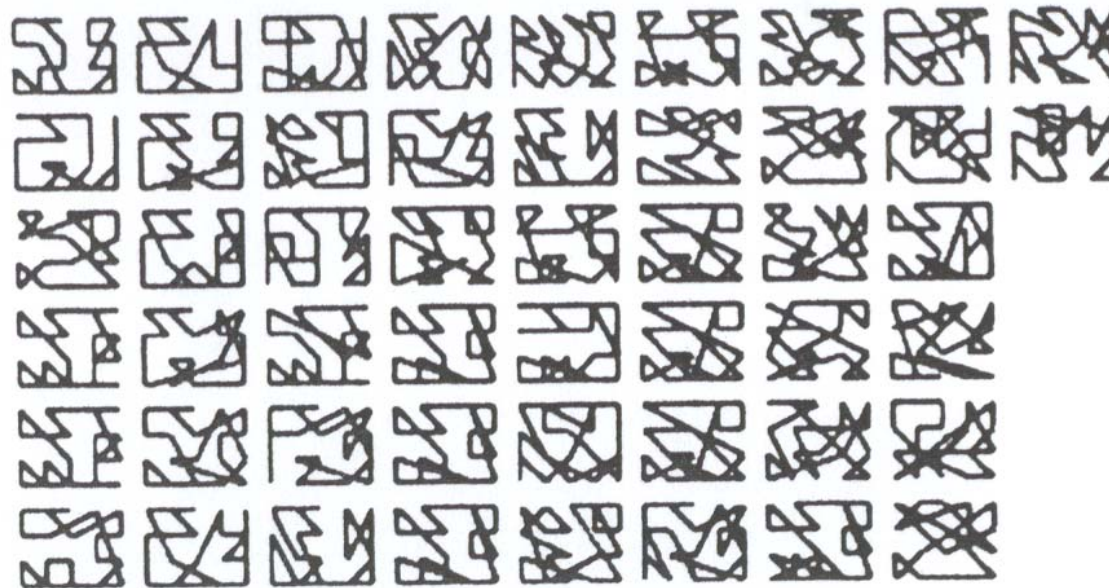


(0)

Visualización de la evolución de una población de 50 cromosomas y 70 iteraciones

# Viajante de Comercio

---

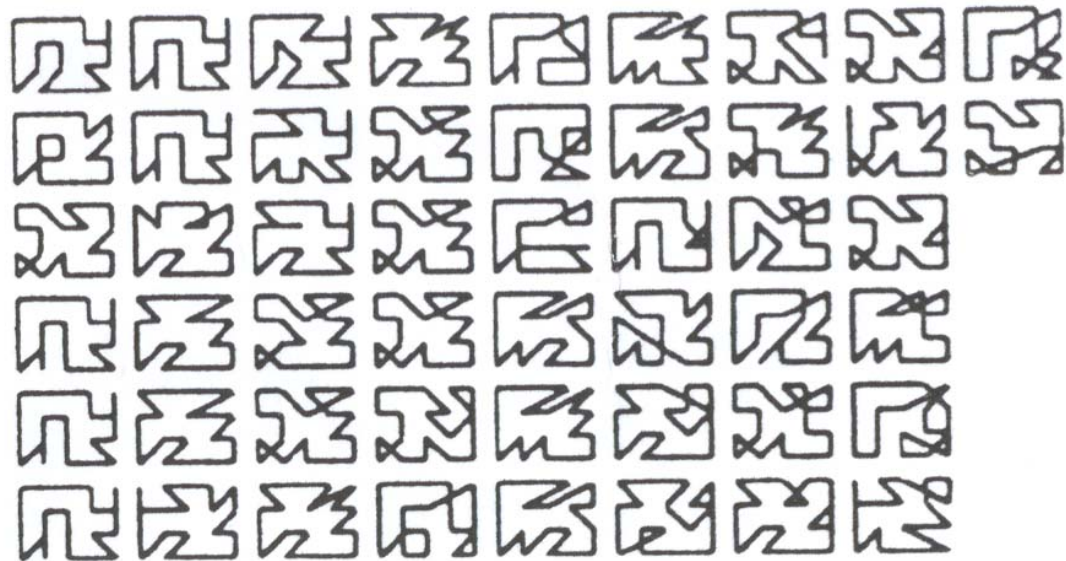


(10)

Visualización de la evolución de una población de 50 cromosomas y 70 iteraciones

# Viajante de Comercio

---

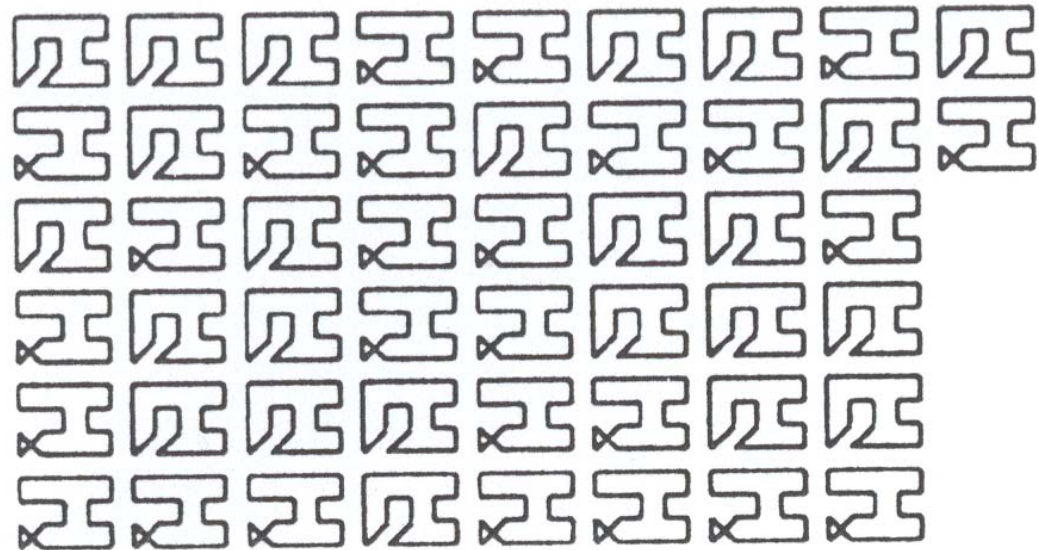


(30)

Visualización de la evolución de una población de 50 cromosomas y 70 iteraciones

# Viajante de Comercio

---



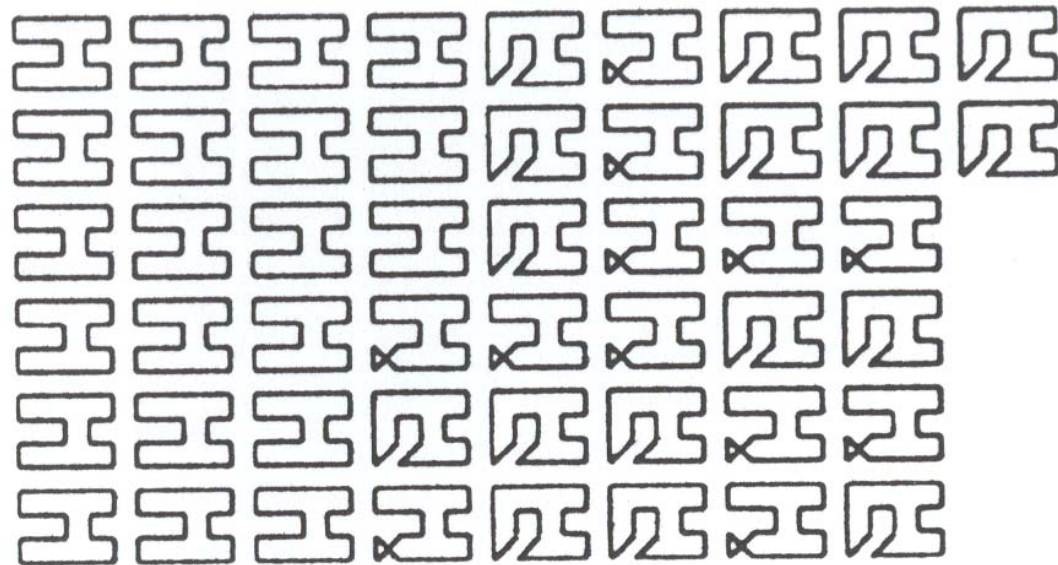
(50)

Visualización de la evolución de una población de 50 cromosomas y 70 iteraciones



# Viajante de Comercio

---

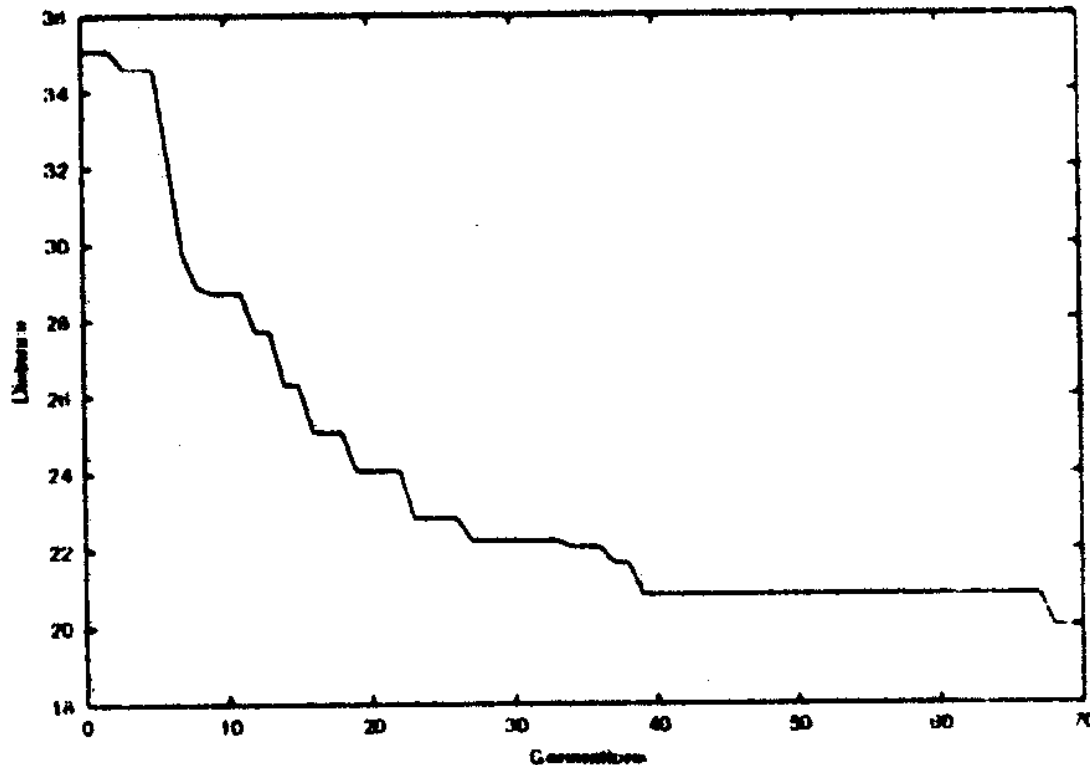


(70)

Visualización de la evolución de una población de 50 cromosomas y 70 iteraciones

# Viajante de Comercio

---



Visualización de la evolución de una población de 50 cromosomas y 70 iteraciones

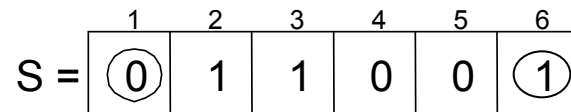


## 8. APLICACIONES: MDP

---

- **Generación de la solución inicial:** aleatoria

- **Esquema de representación:**



- **Selección:** torneo binario
- **Enfoques (2 variantes):** generacional con elitismo (1 individuo) / estacionario
- **Operador de cruce:** Cruce uniforme, probabilidad  $\frac{1}{2}$  para asignar a cada hijo los valores diferentes de un gen en ambos padres.
- **Operador de mutación:** Mutación en un punto seleccionado aleatoriamente, con previa probabilidad de mutación sobre cromosoma.
- **Función objetivo (maximización):** La asociada al problema

# RESUMEN

---

## *Algoritmos Genéticos*

- *basados en una metáfora biológica: evolución*
- *gran potencialidad de aplicación*
- *muy populares en muchos campos*
- *muy potentes en diversas aplicaciones*
- *altas prestaciones a bajo costo*

# RESUMEN

---

## *Algoritmos Genéticos*

*extensión de estudios*

*(Asignatura Bioinformática)*

- *Equilibrio entre diversidad y convergencia*
- *Evolución en paralelo de múltiples soluciones óptimas:*
  - Algoritmos genéticos con nichos*
- *Algoritmos genéticos multiobjetivo*
- *Otros modelos de evolución: Programación genética, estrategias de evolución, evolución diferencial, ...*

# ALGORÍTMICA

## 2012 - 2013

---

- **Parte I. Introducción a las Metaheurísticas**
  - Tema 1. Metaheurísticas: Introducción y Clasificación
- **Parte II. Métodos Basados en Trayectorias y Entornos**
  - Tema 2. Algoritmos de Búsqueda Local Básicos
  - Tema 3. Algoritmos de Enfriamiento Simulado
  - Tema 4. Algoritmos de Búsqueda Tabú
  - Tema 5. Métodos Basados en Trayectorias Múltiples I: Métodos Multiarranque Básicos y GRASP
  - Tema 6. Métodos Basados en Trayectorias Múltiples II: ILS y VNS
- **Parte III. Métodos Basados en Poblaciones**
  - Tema 7. Algoritmos Genéticos
- **Parte IV. Intensificación y Diversificación**
  - **Tema 8. Estudio del Equilibrio entre Intensificación y Diversificación**
- **Parte V. Metaheurísticas Híbridas: Poblaciones y Trayectorias**
  - Tema 9. Algoritmos Meméticos
  - Tema 10. *Scatter Search*
- **Parte VI. Paralelización de Metaheurísticas**
  - Tema 11. Metaheurísticas en Sistemas Descentralizados
- **Parte VII. Conclusiones**
  - Tema 12. Algunas Consideraciones sobre la Adaptación de Metaheurísticas a la Resolución de Problemas