

# ALGORITMOS GENÉTICOS

Oscar Cordón, Universidad de Granada

- 
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. PARALELIZACIÓN DE LOS ALGORITMOS GENÉTICOS
  9. APLICACIONES

# BIBLIOGRAFÍA

---

- D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley, 1989.**
- Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs. Springer Verlag, 1996.**
- T. Bäck, D.B. Fogel, Z. Michalewicz, Handbook of Evolutionary Computation, Oxford Univ. Press, 1997.**

# 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**

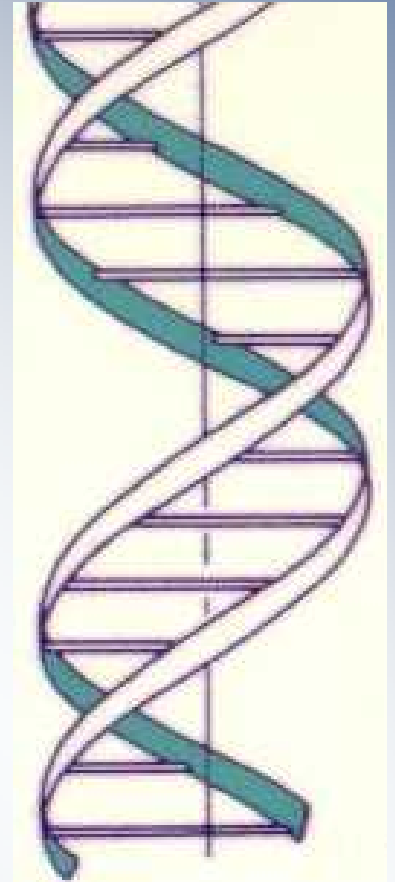


# ¿Qué es un Algoritmo Genético?

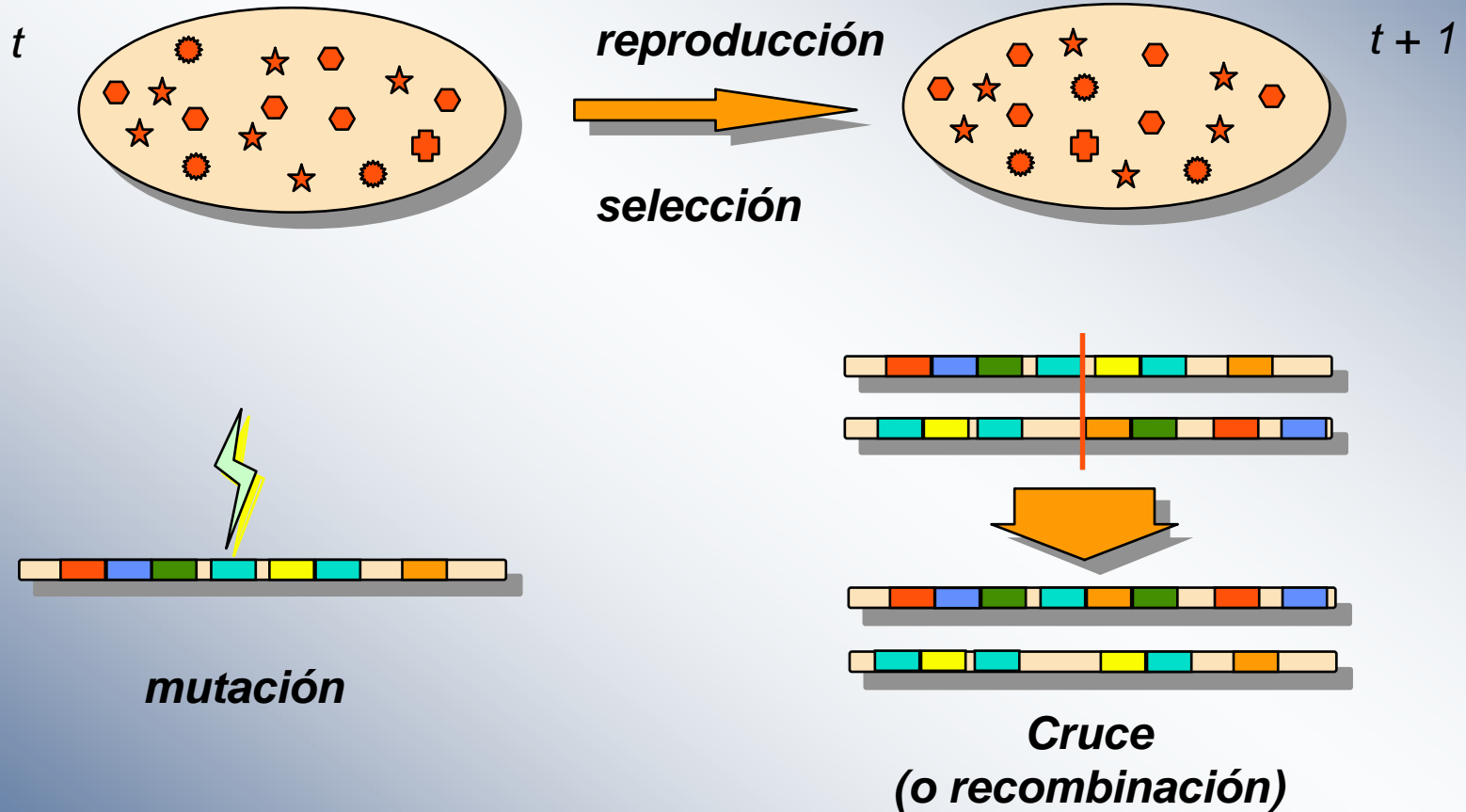
## Los Algoritmos Genéticos

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

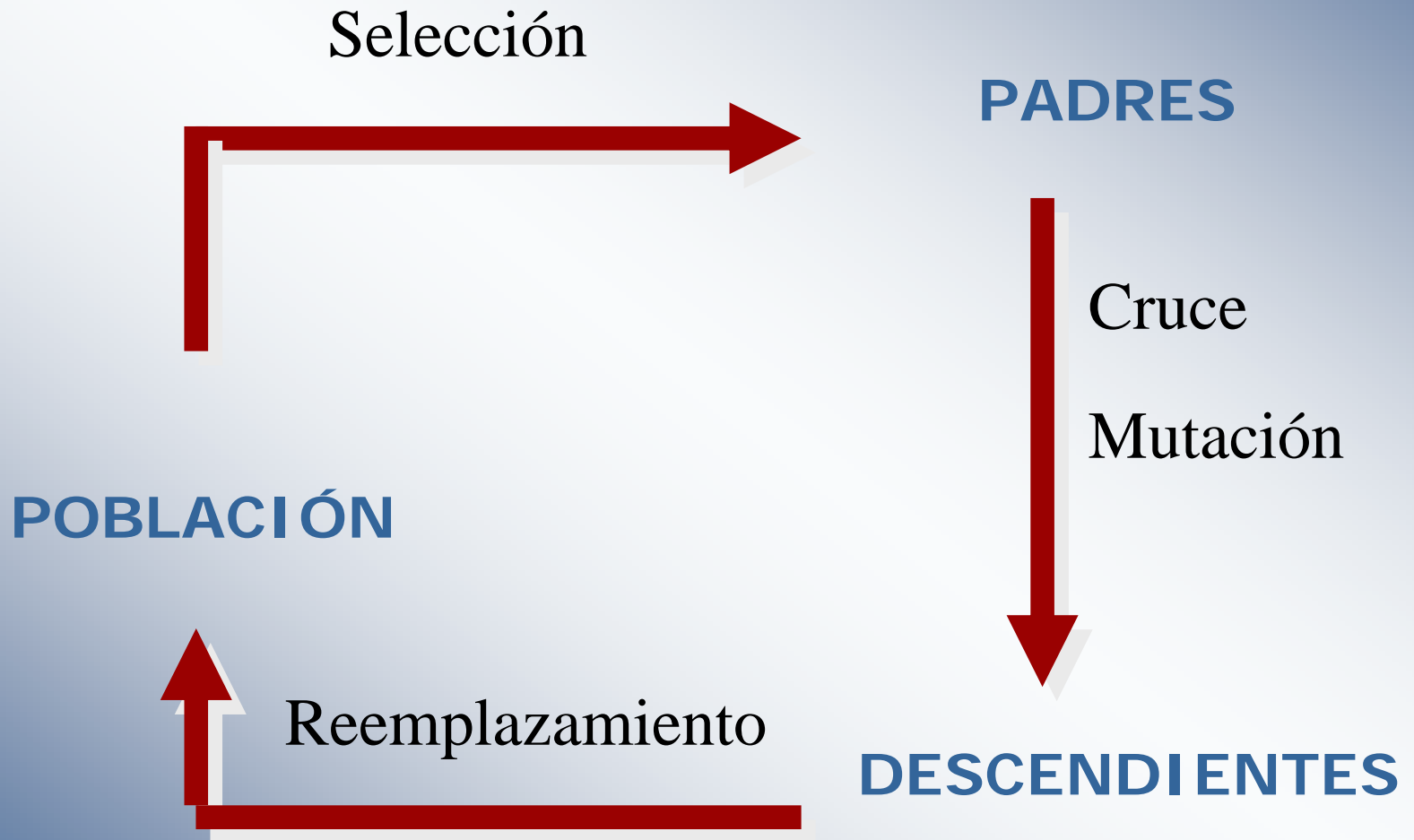
**Evolución Natural  
y  
Evolución Genética**



# Los Ingredientes



# El ciclo de la Evolución



# Estructura de un Algoritmo Genético

## Algoritmo Genético Básico

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  $P(t-1)$

$P''(t) \leftarrow$  cruce  $P'(t)$

$P'''(t) \leftarrow$  mutación  $P''(t)$

$P(t) \leftarrow$  reemplazamiento  $(P(t-1), P'''(t))$

evaluar  $P(t)$

Final(2)

Final(1)

## 2. ¿CÓMO SE CONSTRUYE?

---

### 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

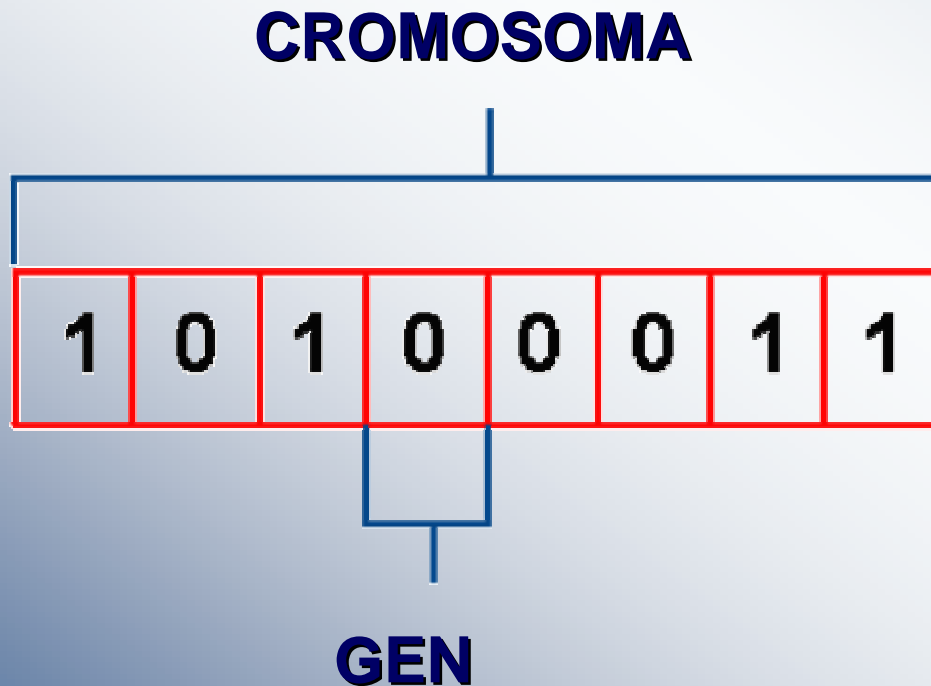
# Representación

---

- Debemos disponer de un mecanismo para codificar un individuo como un genotipo
- Existen muchas maneras de hacer esto y se debe elegir la más relevante para el problema en cuestión
- Una vez elegida una representación, tenemos que tener en mente cómo los genotipos (codificación) serán evaluados y qué operadores genéticos habrá 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

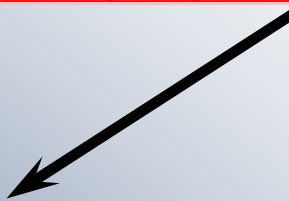
El fenotipo puede ser números enteros

**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 puede ser números reales

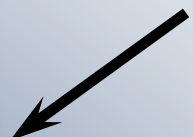
Ejemplo: un número entre 2,5 y 20,5 utilizando 8 dígitos binarios

**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
- 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

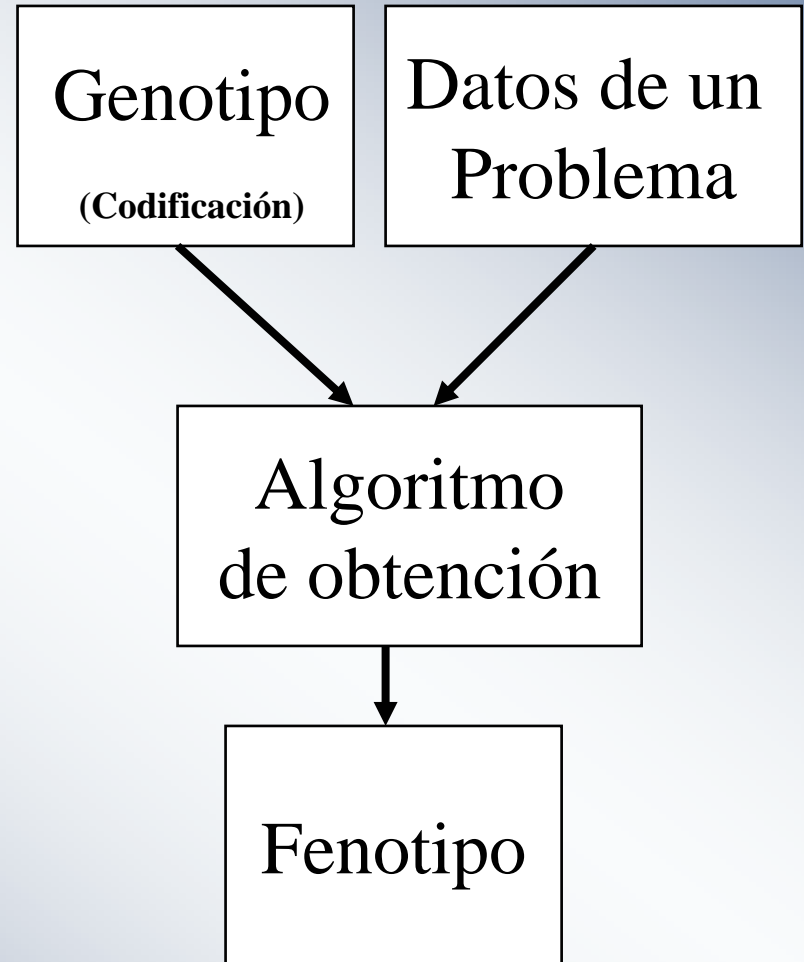
# Inicialización

---

- 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: *fitness* o valor de adecuación

- 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, estas se pueden introducir en el costo como penalización
- Con múltiples objetivos se busca una solución de compromiso

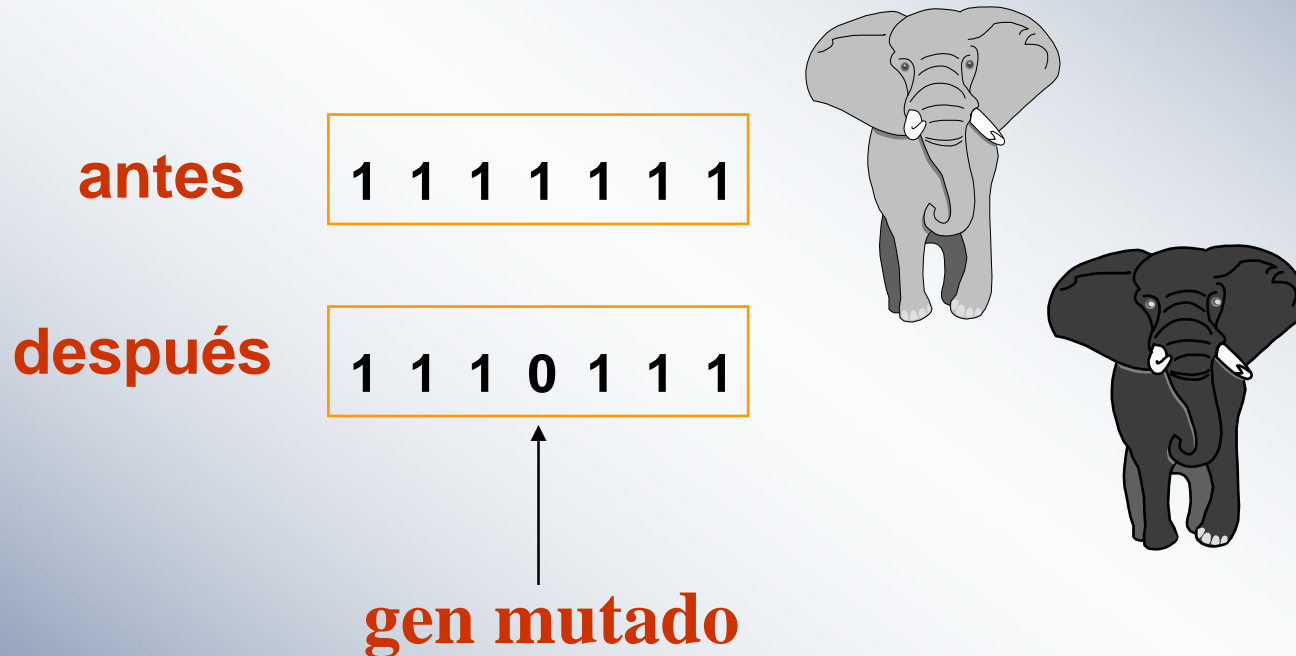


# Operador de mutación

---

- 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

# Ejemplo: Mutación para representación discreta binaria



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

Se seleccionan aleatoriamente dos genes y se intercambian

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



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

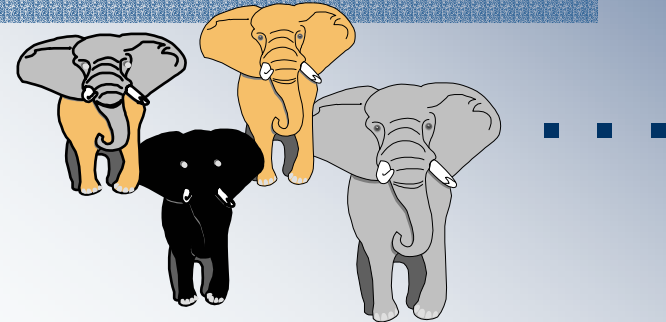
# Operador de Cruce

---

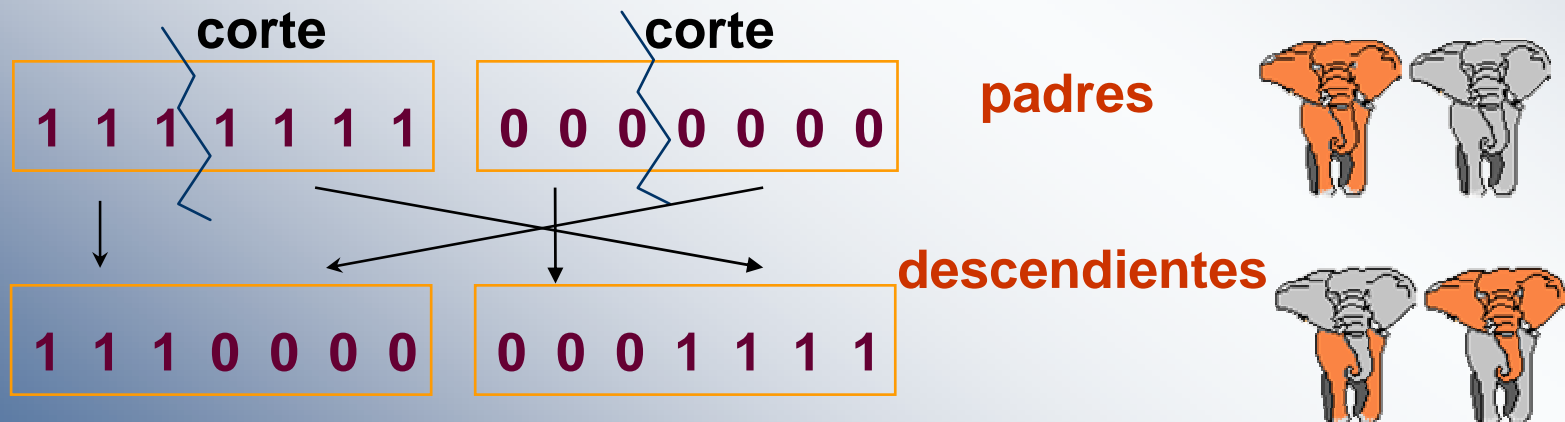
- Podríamos tener uno o más operadores de cruce para nuestra representación
- Algunos aspectos importantes a tener en cuenta son:
  - Los hijos deberían heredar algunas características de **cada** padre. Si éste no es el caso, entonces estamos ante un operador de mutación
  - Se debe diseñar de acuerdo a la representación
  - La recombinación debe producir cromosomas válidos

# Ejemplo: Operador de cruce para representación binaria

Población:



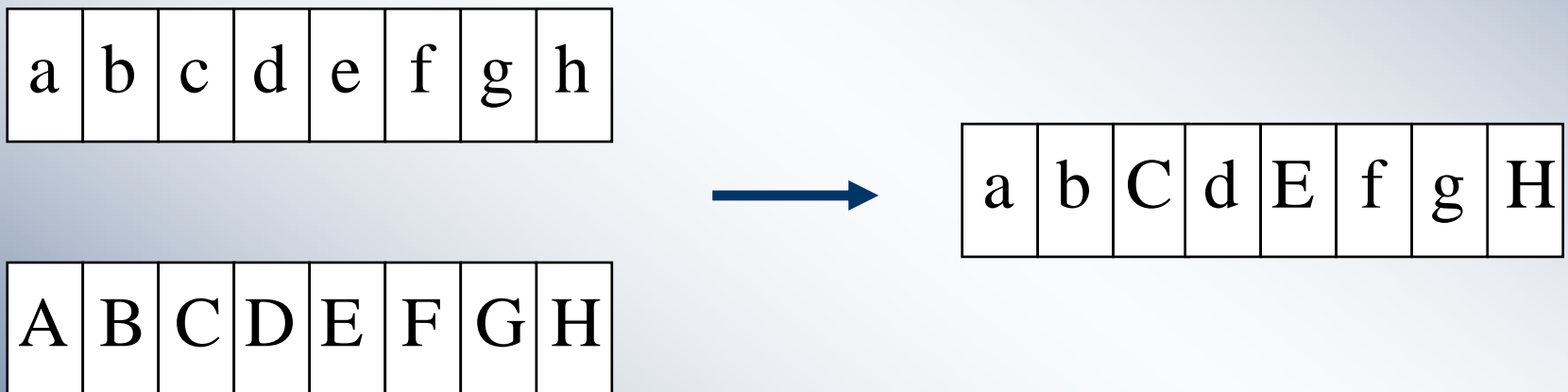
Cada cromosoma se corta en  $n$  partes que son recombinadas. (Ejemplo para  $n = 1$ )



# Ejemplo: Operador de cruce para representación real

---

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



Existen muchos operadores específicos para la codificación real

# Ejemplo: Operador de cruce para representación real

Recombinación aritmética (cruce aritmético):

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

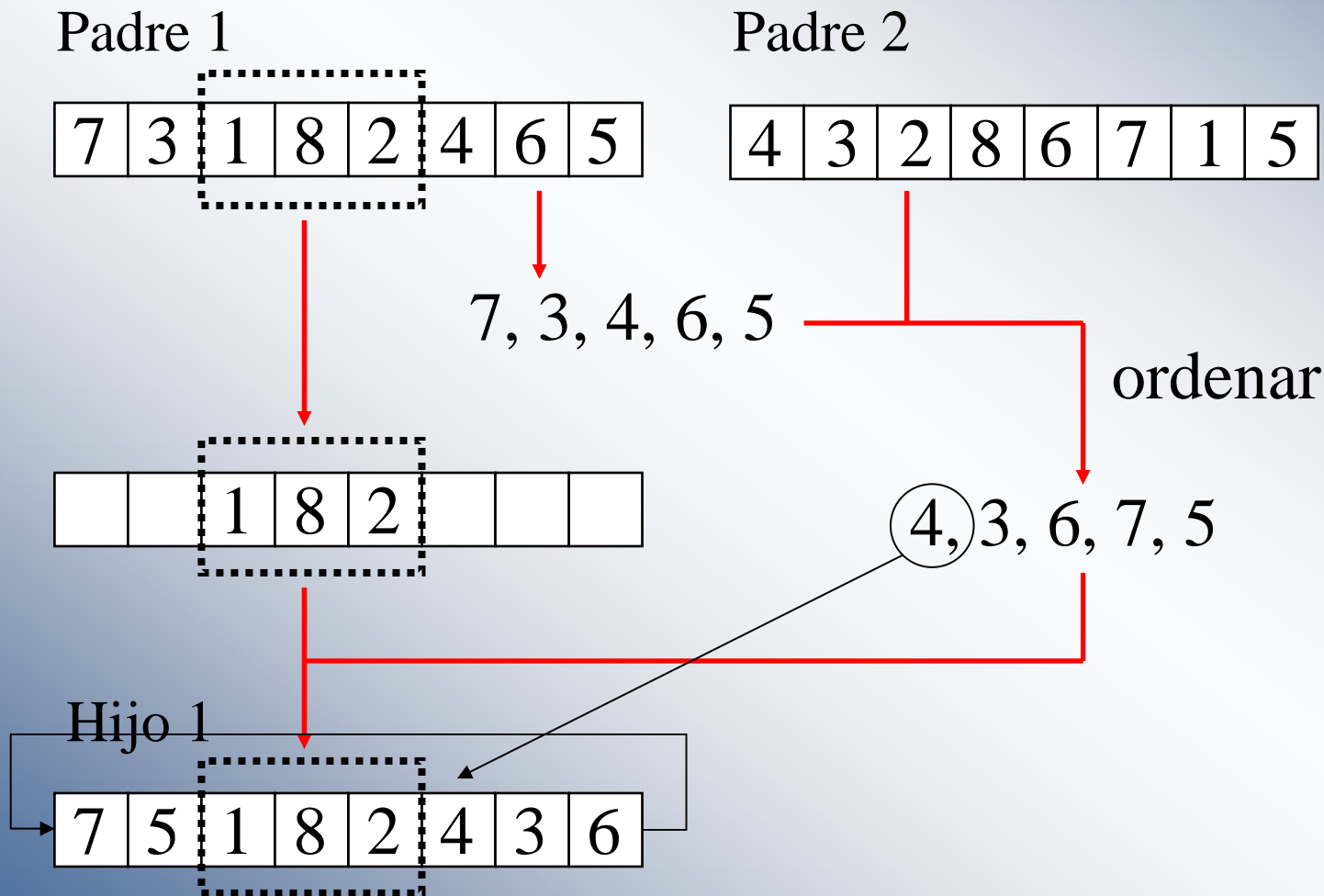
A	B	C	D	E	F
---	---	---	---	---	---



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



# Ejemplo: Operador de cruce para representación de orden (Cruce OX)



# Estrategia de Selección

---

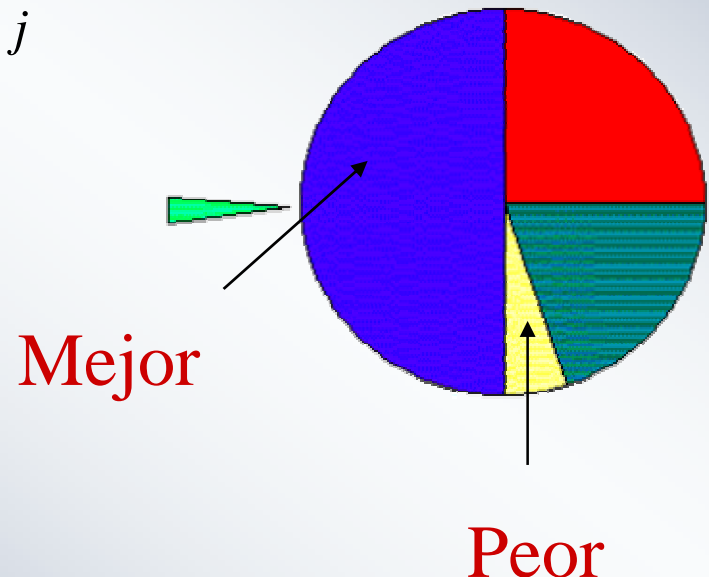
- Debemos garantizar que los mejores individuos tengan una mayor posibilidad de ser padres (reproducirse) frente a los individuos menos buenos
- Debemos ser cuidadosos para dar una posibilidad de reproducirse a los individuos menos buenos. Éstos pueden incluir material genético útil en el proceso de [reproducción](#)
- Esta idea nos define la **presión selectiva** que conducirá la reproducción como la selección fuerte de los mejores

# Ejemplo: Selección proporcional

- El número de veces que un individuo se debe reproducir es

$$P(S_i) = \frac{f_i}{\sum_j f_j}$$

- Los mejores individuos tienen:
  - Más espacio en la ruleta
  - Más probabilidad de ser seleccionados



# Ejemplo: Selección proporcional

---

## Desventajas:

- Peligro de convergencia prematura porque los mejores individuos dominan la población muy rápidamente
- Baja presión selectiva cuando los valores de la función objetivo están muy cercanos
- Comportamientos diferentes cuando se realizan traslaciones sobre la función de evaluación

# Ejemplo: Selección proporcional

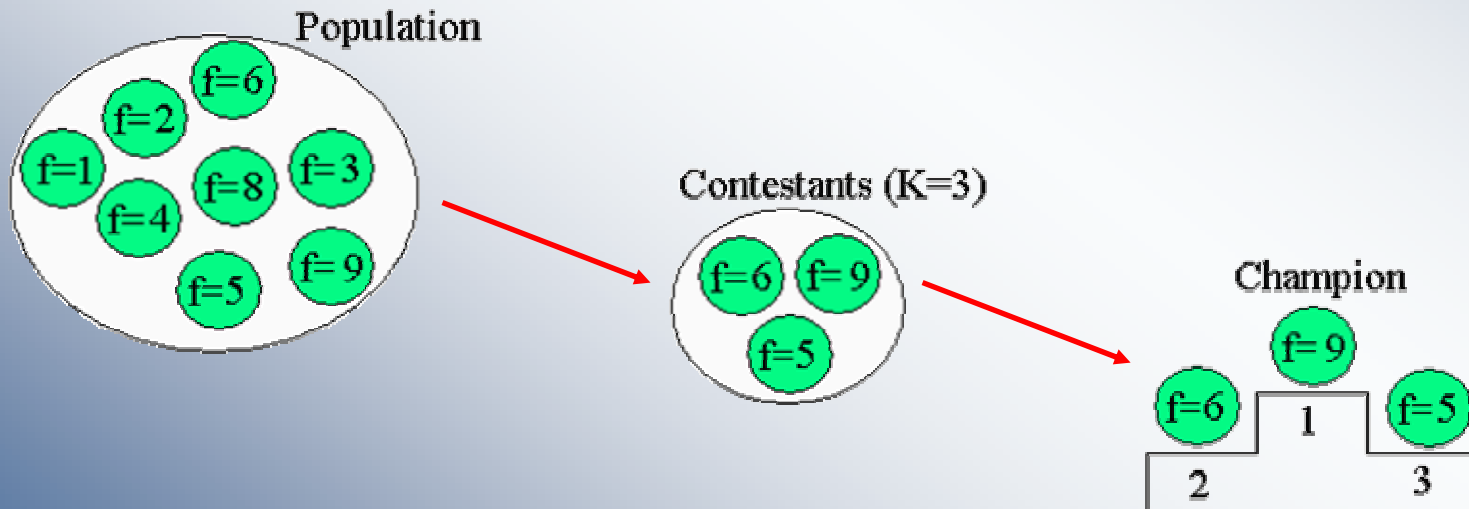
---

## Escalado del fitness (valor de adecuación): Una solución

- Ajustar el fitness a un rango:
  - Ej. Rangos de 0 a 1
- Normalizar:
  - La suma de los fitness igual a 1

# Ejemplo: Selección por torneo

- Seleccionar  $k$  individuos aleatoriamente, sin reposición
- Coger el mejor
  - $k$  se llama el tamaño del torneo



# Ejemplo: Selección basada en orden

---

- Los individuos se ordenan por su valor de función objetivo de mejor a peor. El lugar ocupado en la lista se llama **el orden del cromosoma**
- En lugar del valor de la función objetivo, se utiliza el orden del cromosoma, para ordenar entre un máximo y un mínimo

# Ejemplo: Selección basada en orden

- Función objetivo:  $f(A) = 5, f(B) = 2, f(C) = 19$
- Orden:  $r(A) = 2, r(B) = 3, r(C) = 1$

$$h(x) = \min + (\max - \min) * \frac{(r(x) - 1)}{n - 1}$$

- Función:  $h(A) = 3, h(B) = 5, h(C) = 1$
- Proporción en la ruleta:

$$p(A) = 11,1\%, p(B) = 33,3\%, p(C) = 55,6\%$$



# Estrategia de Reemplazamiento

---

- La presión selectiva se ve también afectada por la forma en que los cromosomas de la población son reemplazados por los nuevos descendientes
- Podemos utilizar métodos de reemplazamiento aleatorios, o determinísticos
- Podemos decidir no reemplazar al mejor(es) cromosoma(s) de la población: **Elitismo**

# 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

# Componentes



### 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

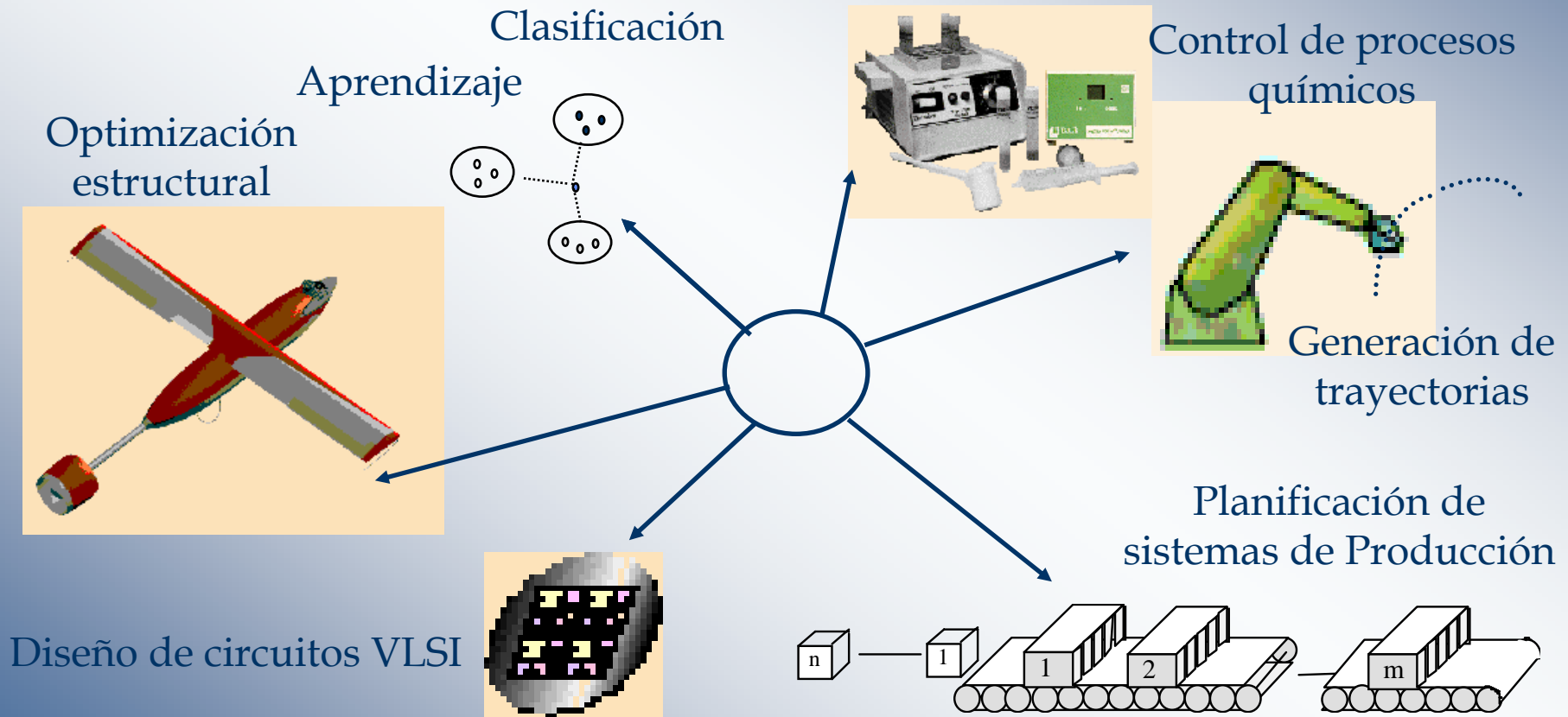
**La nueva población reemplaza directamente a la antigua**

**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*

# 6. DOMINIOS DE APLICACIÓN





# 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.

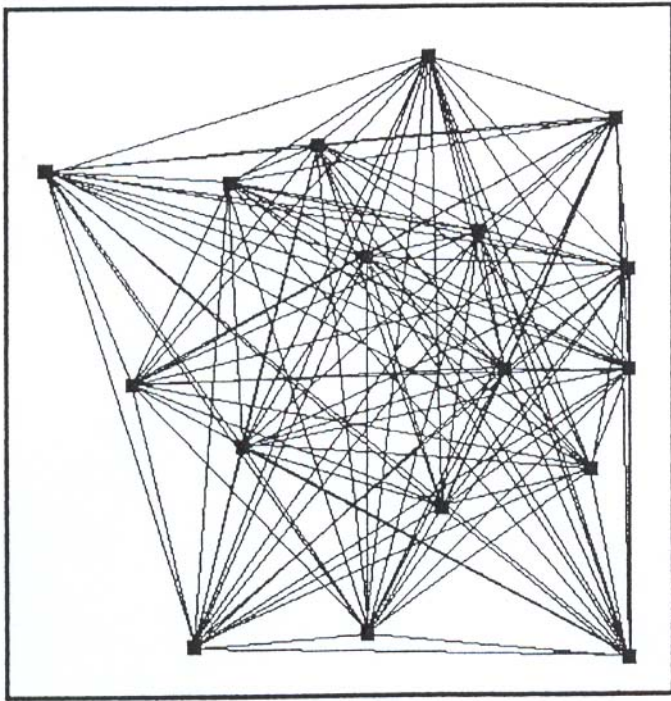
Población: 61 cromosomas - Elitismo

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

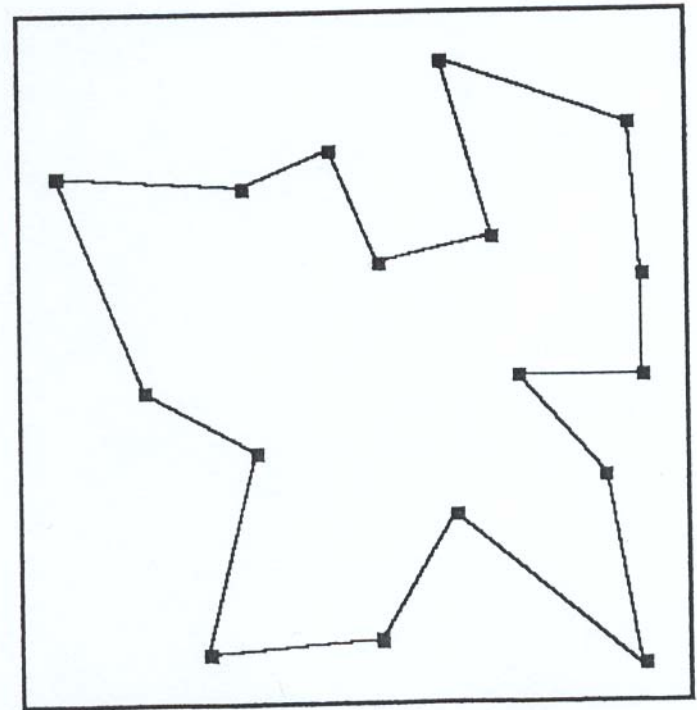
Mutación: Inversión de una lista ( $P_m = 0,01$  – cromosoma)

# Viajante de Comercio

---

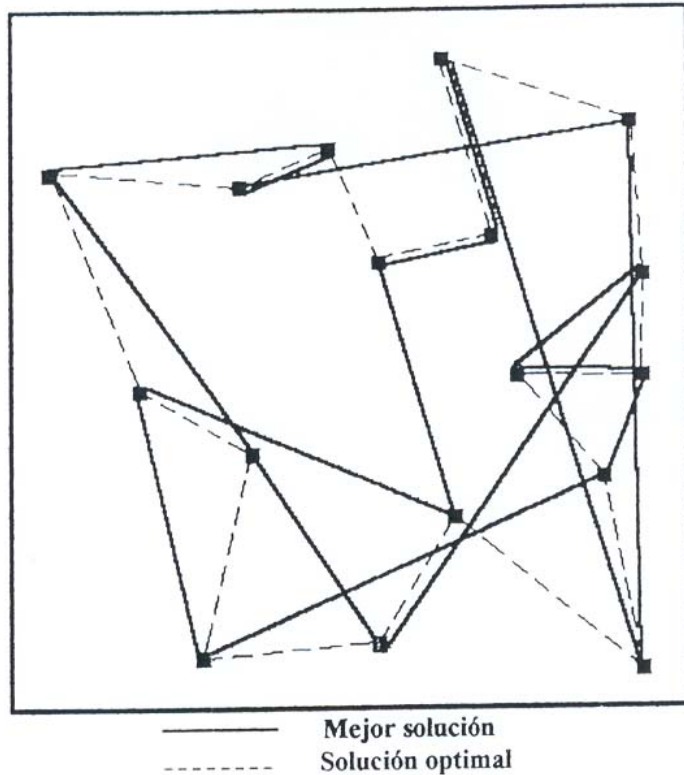


$17! = 3.5568743 \text{ e}14$  recorridos posibles

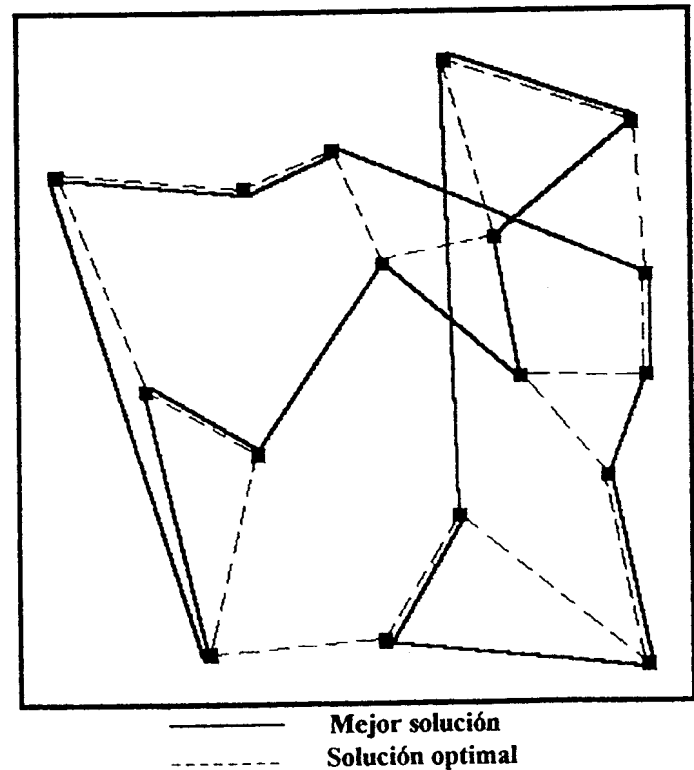


Solución óptima: 226.64

# Viajante de Comercio



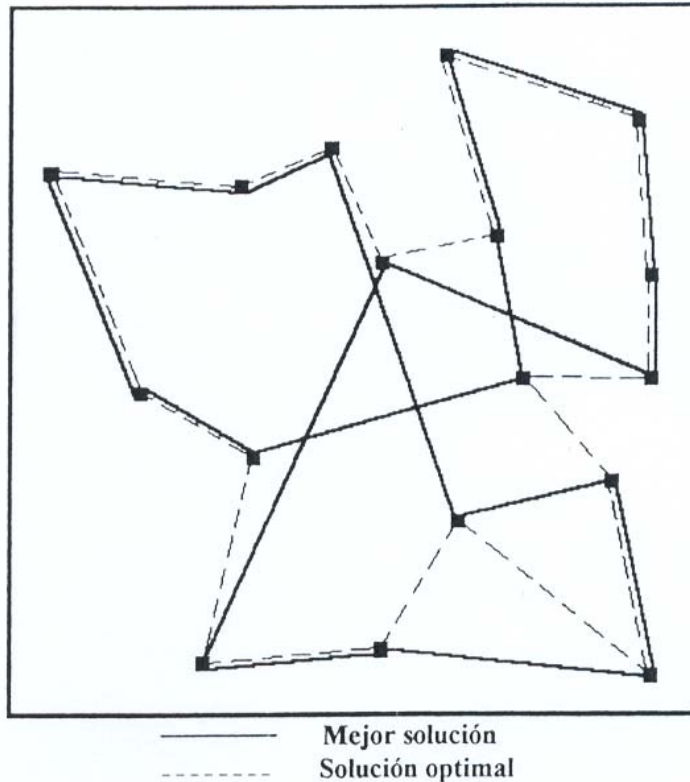
Iteración: 0 Costo: 403.7



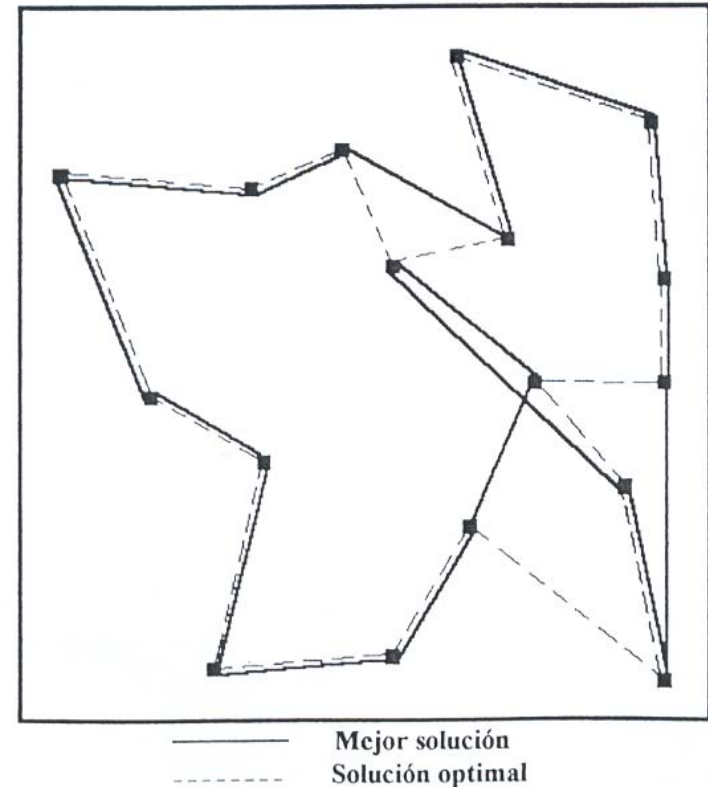
Iteración: 25 Costo: 303.86

Solución óptima: 226.64

# Viajante de Comercio



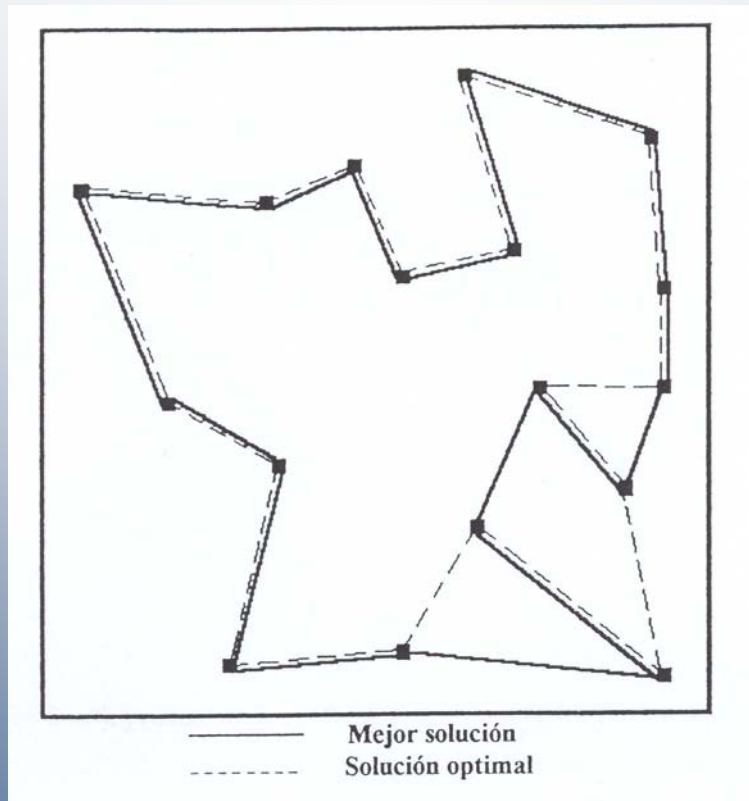
Iteración: 50 Costo: 293,6



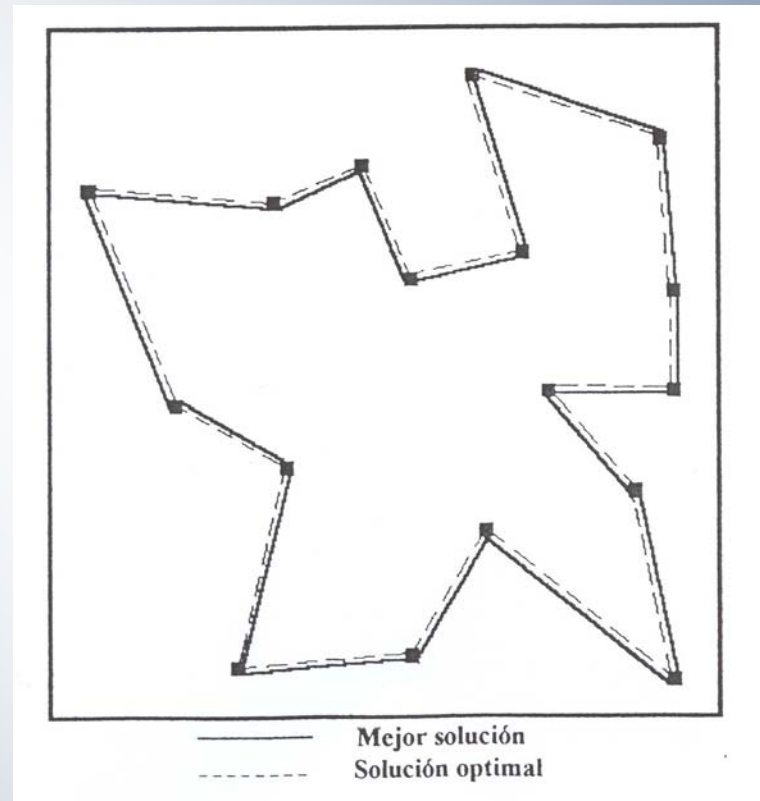
Iteración: 100 Costo: 256,55

Solución óptima: 226,64

# Viajante de Comercio

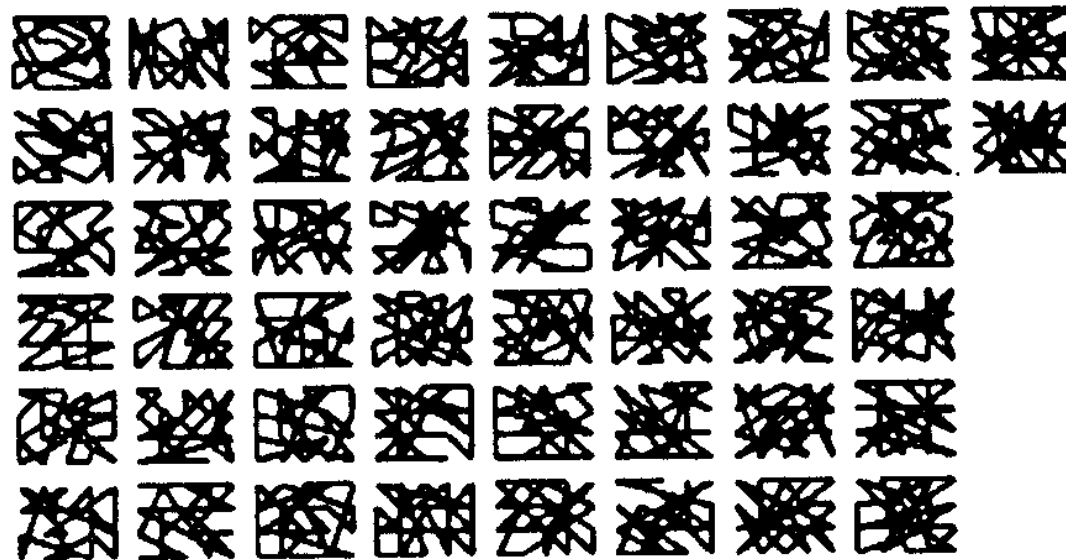


Iteración: 200 Costo: 231,4



Iteración: 250 Solución  
óptima: 226,64

# Viajante de Comercio

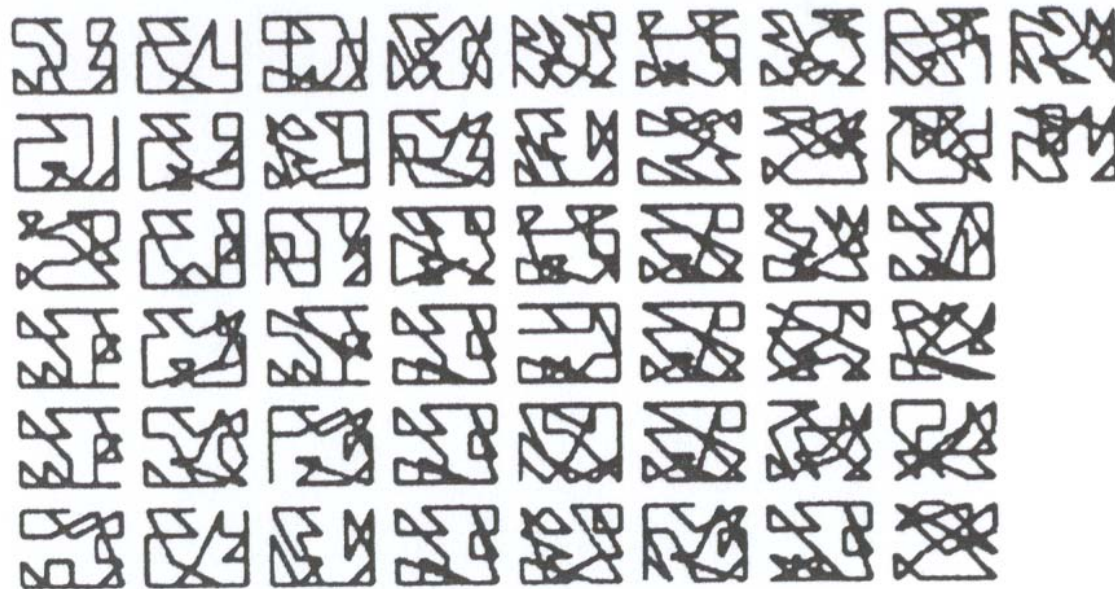


(0)

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



# Viajante de Comercio

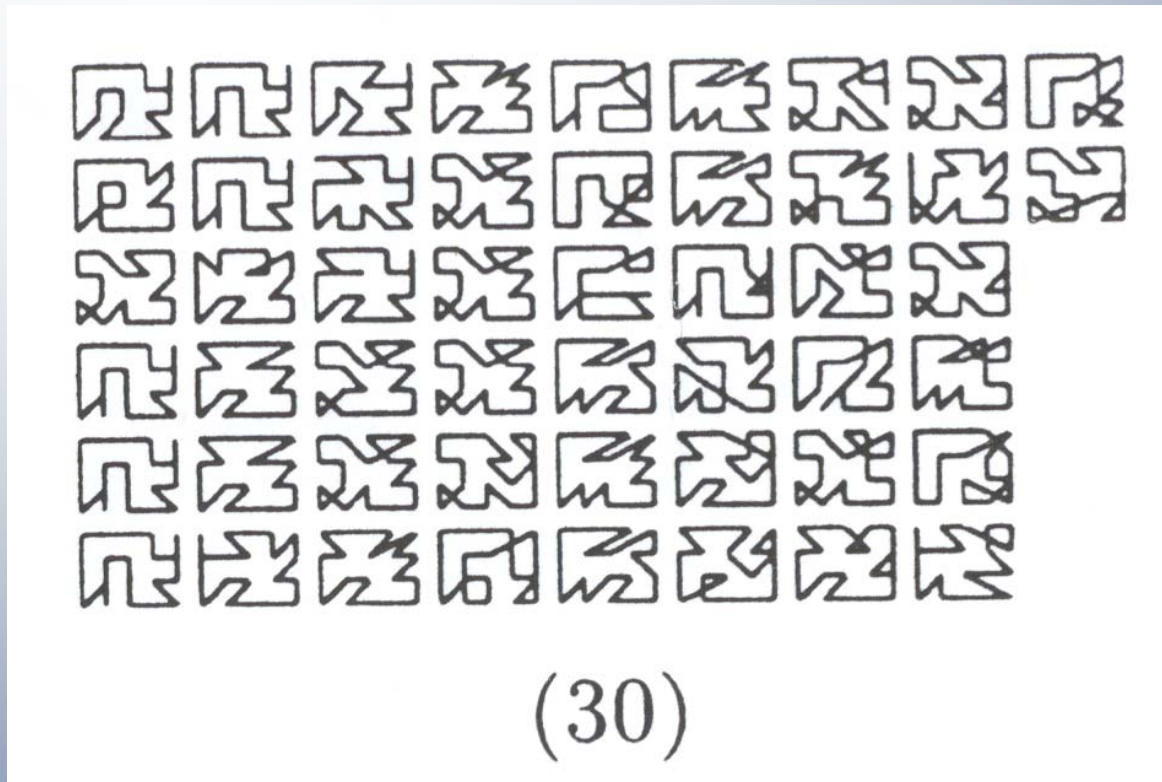


(10)

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



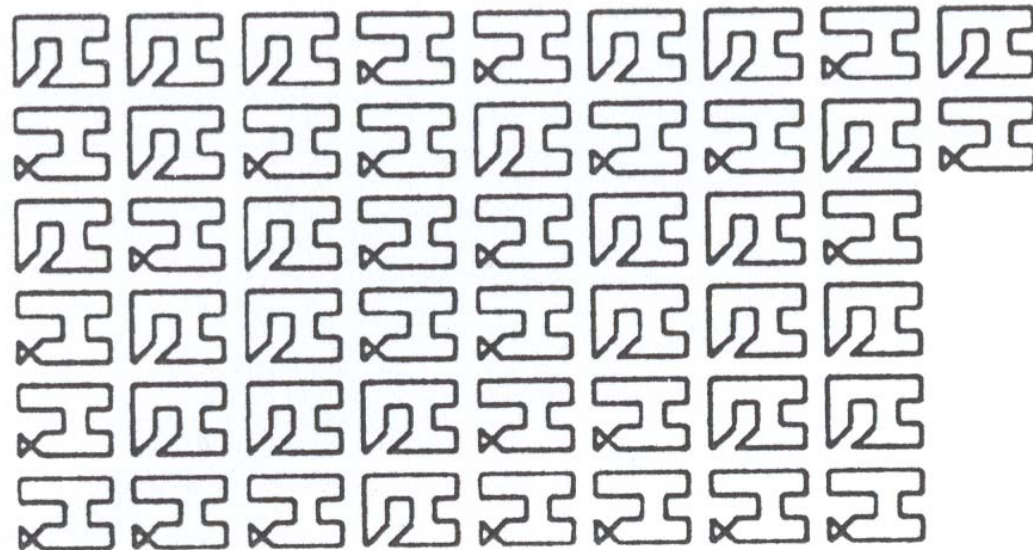
# Viajante de Comercio



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

# Viajante de Comercio

---

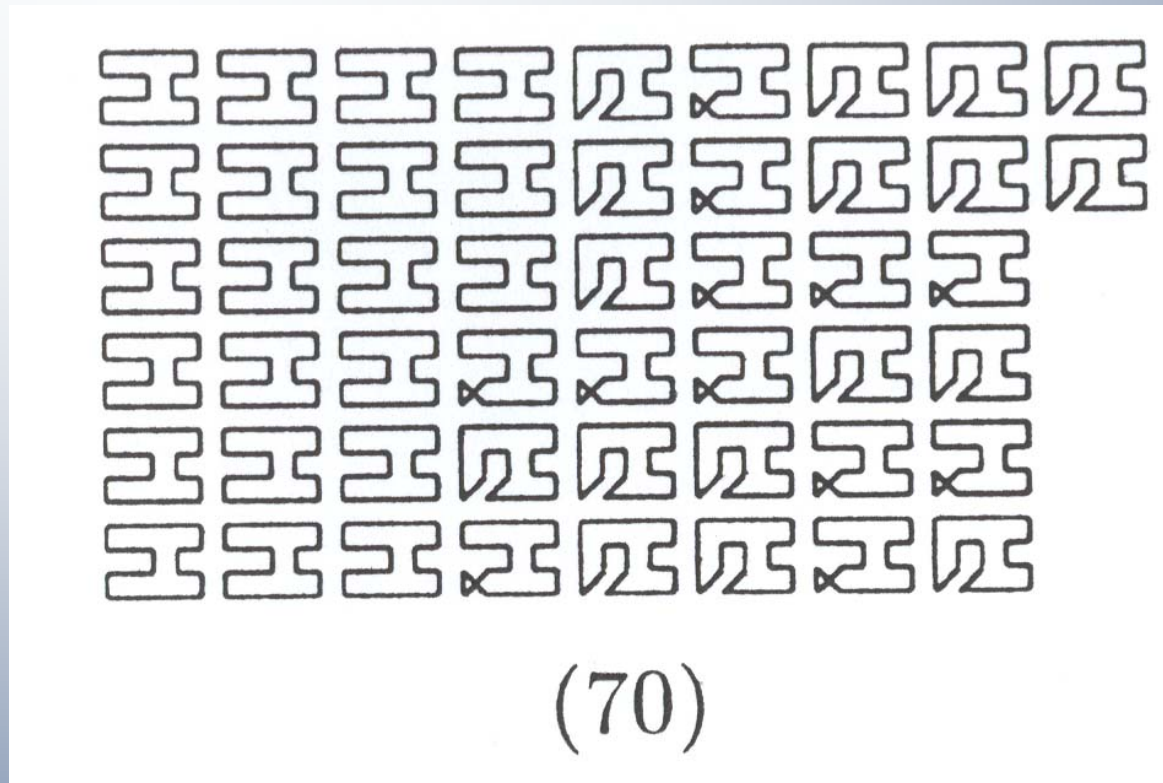


(50)

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

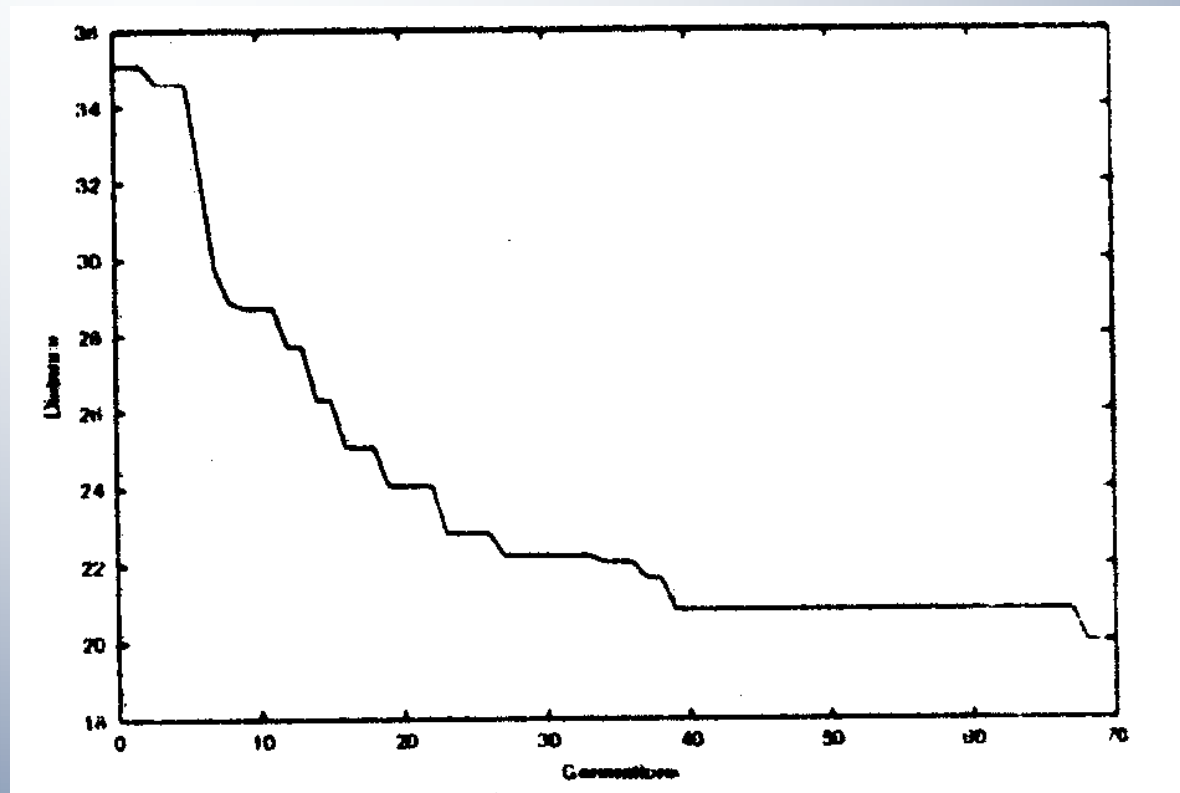
# Viajante de Comercio

---



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. PARALELIZACIÓN DE LOS AGs

---

### OBJETIVOS (Tema 1)

1. Preservar la calidad de las soluciones reduciendo el tiempo de ejecución
2. Incrementar la calidad de las soluciones sin aumentar el tiempo de cálculo:
  - Aumentando las iteraciones con una paralelización efectiva
  - Ventajas de un diseño paralelo ejecutado secuencialmente, que permita introducir mayor diversidad en el proceso de búsqueda y que evite la convergencia prematura

## 8. PARALELIZACIÓN DE LOS AGs

---

- Los AGs son muy adecuados para una paralelización efectiva dado que evolucionan una población de soluciones en paralelo
- No son directamente paralelizables ya que es necesario (al menos en selección y cruce) un control global
- Consideraciones a tener en cuenta:
  - Sincronización de las operaciones
  - Arquitecturas de comunicación
  - Separación entre el paralelismo de los algoritmos y sus implementaciones en diferentes plataformas

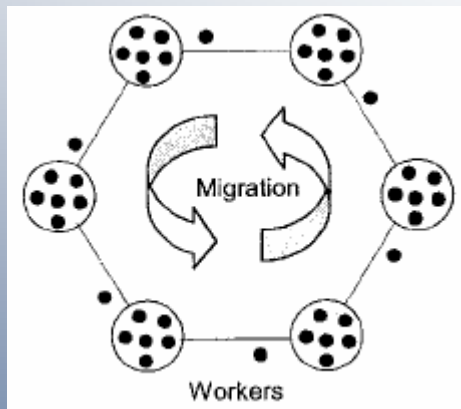


# 8. PARALELIZACIÓN DE LOS AGs

## Tipos de Descentralización

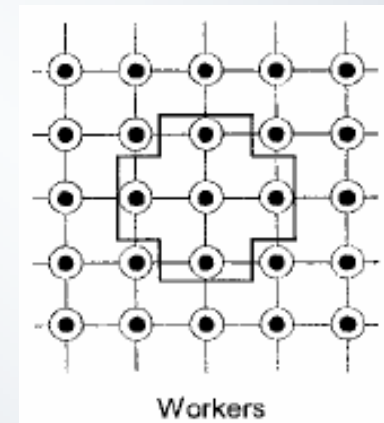
### ■ Distribuidos

- Se definen subpoblaciones
- Comunicación mediante intercambio de individuos



### ■ Celulares

- Sólo hay una población
- Comunicación mediante vecindad de individuos



# Modelos Distribuidos: Modelo de Isla

## Fundamento

---

- En entornos aislados, tales como las islas, se encuentran especies animales que se adaptan más eficazmente a las peculiaridades de su entorno que las correspondientes a superficies de mayor amplitud, esto ha dado lugar a los llamados nichos

### Hipótesis

- La competición entre varias subpoblaciones podría proporcionar una búsqueda más efectiva que la evolución de una gran población en la que todos los miembros coexistieran

### Propuesta

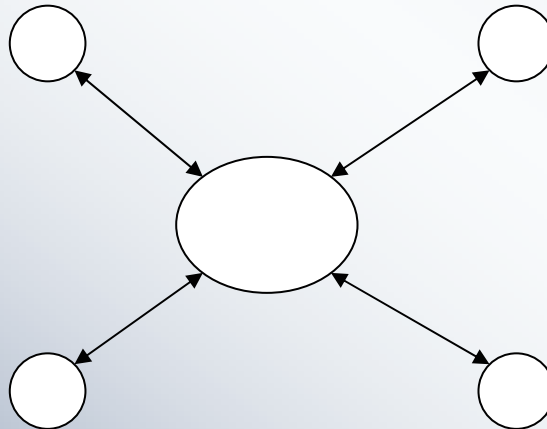
- **Modelo Isla:** Tener varias poblaciones aisladas que evolucionan en paralelo y periódicamente intercambian por migración sus mejores individuos con las subpoblaciones vecinas



# Modelos Distribuidos: Modelo de Isla

## Estructuras de Intercomunicación

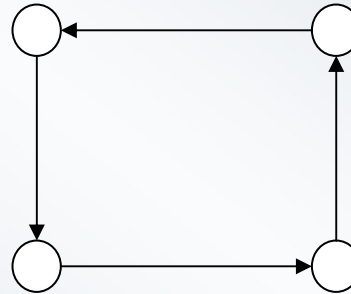
- **Estrella:** la subpoblación con mayor promedio objetivo se selecciona como maestra y las demás como subordinadas. Todas las subpoblaciones subordinadas envían sus mejores individuos a la maestra, y a su vez, ésta envía también sus mejores individuos a cada una de las subordinadas



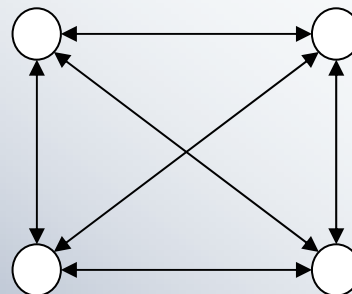
# Modelos Distribuidos: Modelo de Isla

## Estructuras de Intercomunicación

- **Anillo:** Cada subpoblación envía sus mejores individuos a la subpoblación vecina más próxima en un único sentido de flujo



- **Red:** Todas las subpoblaciones envían sus mejores individuos a todas las demás



# Modelos Celulares o Masivamente Paralelos

## Fundamento

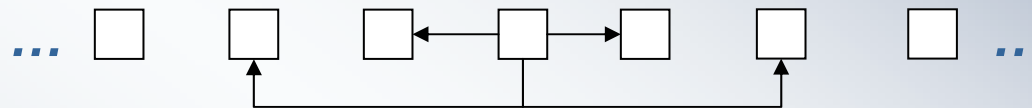
---

- Se trabaja con una única población y se pretende paralelizar las operaciones que realiza un AG clásico
- Cada individuo es colocado en una celda de un plano cuadriculado. La selección y el cruce se aplican entre individuos vecinos sobre la cuadrícula de acuerdo a una estructura de vecinos preestablecida
- **Función de evaluación:** Cada procesador elemental debe tener acceso sólo a aquellos individuos para los que calculará su función de evaluación
- **Cruce:** Cada procesador elemental que cree un nuevo individuo debe tener acceso a todos los otros individuos puesto que cada uno de ellos se puede seleccionar como padre
- **Mutación:** Cada procesador elemental necesita sólo los individuos con los que trate

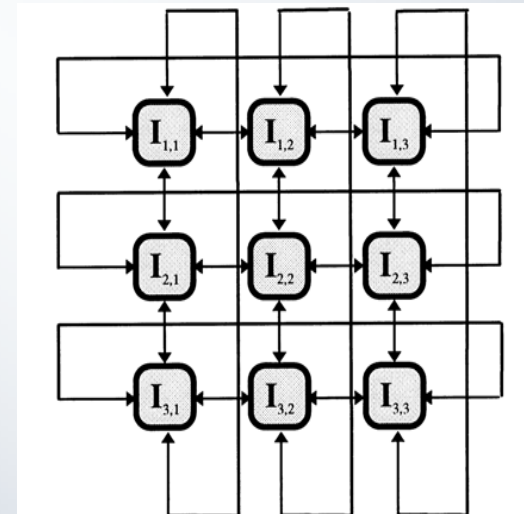
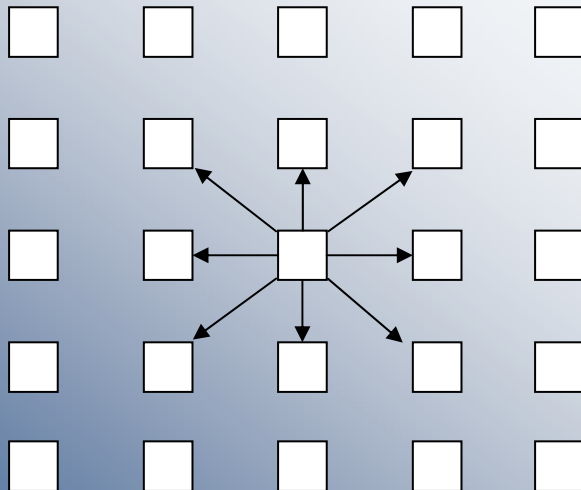
# Modelos Celulares o Masivamente Paralelos

## Estructuras de Intercomunicación

### ■ Lista Circular

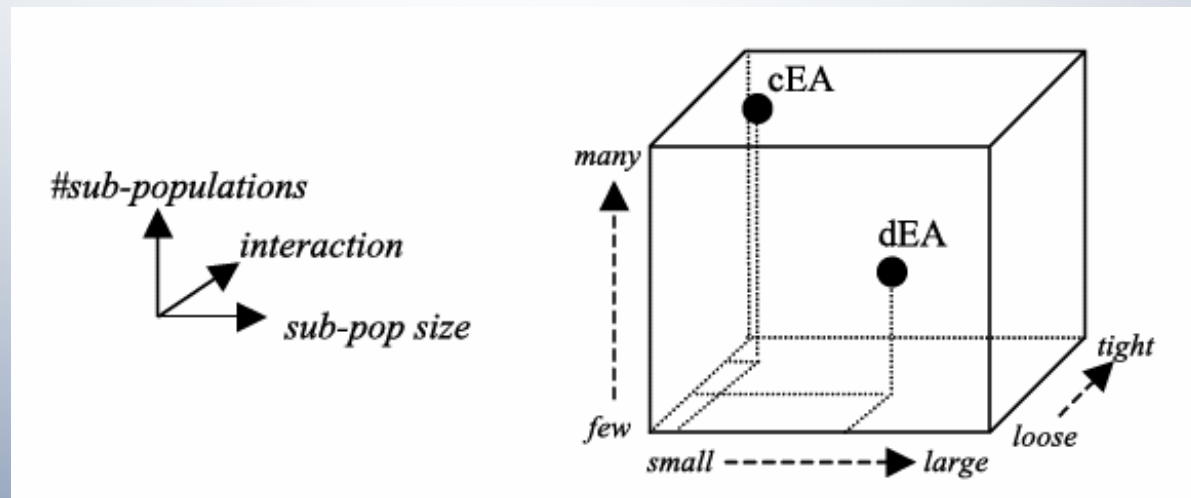


### ■ Matriz bidimensional



# Relación entre Modelos Distribuidos y Celulares

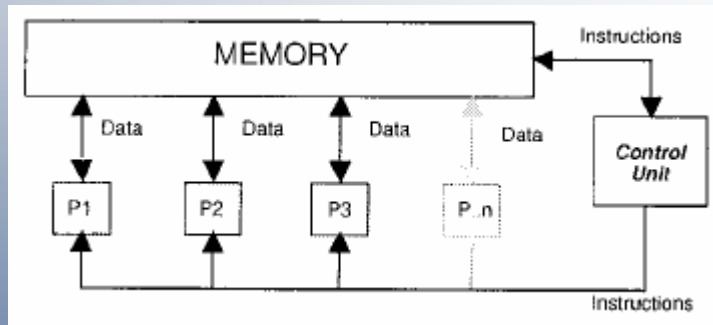
- dEA = Modelos Distribuidos
- cEA = Modelos Celulares



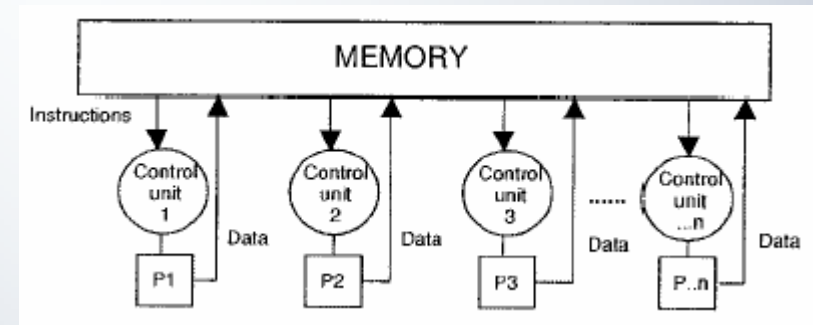
# Plataformas *Hardware*

- Generalmente, los modelos distribuidos se implementan sobre arquitecturas paralelas MIMD mientras que los celulares sobre SIMD

## SIMD



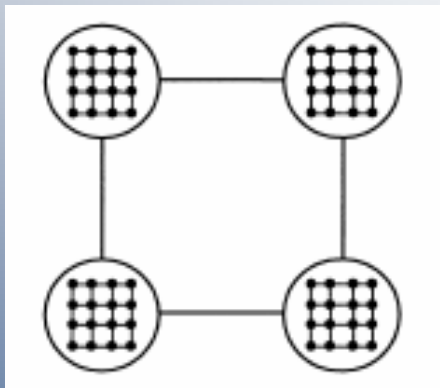
## MIMD



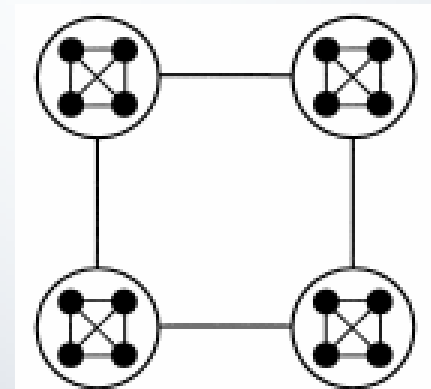
## Otras Cuestiones

- Plataformas homogéneas o heterogéneas
- Comunicación síncrona o asíncrona
- Hibridaciones según sean subpoblaciones de grano (número de individuos por procesador) grueso o fino

Grano grueso y  
grano fino



Grano grueso y  
grano grueso



## 9. APLICACIONES

### 9.1. Viajante de Comercio

---

1. **Generación de la población inicial:** aleatoria
2. **Representación:** permutación  $\{1, \dots, n\}$
3. **Selección:** torneo binario
4. **Enfoques (2 variantes):** generacional con elitismo (1 individuo) / estacionario
5. **Mutación:** operador 2-opt
6. **Cruce:** Elegir dos distintos entre OX (transparencia 34), PMX y CX



## 9. APLICACIONES

### 9.1. Viajante de Comercio

#### Cruce para representación de orden PMX

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

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

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

$$\text{Hijo}'_1 = (x \ x \ x \mid 1 \ 8 \ 7 \ 6 \mid x \ x)$$

$$\text{Hijo}'_2 = (x \ x \ x \mid 4 \ 5 \ 6 \ 7 \mid x \ x)$$

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

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

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

## 9. APLICACIONES

### 9.1. Viajante de Comercio

#### Cruce para representación de orden CX

- Partiendo de la ciudad  $i$  del primer padre, CX toma la siguiente ciudad  $j$  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

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

## 9. APLICACIONES

### 9.2. *Free Disposal Hull* Centralizado

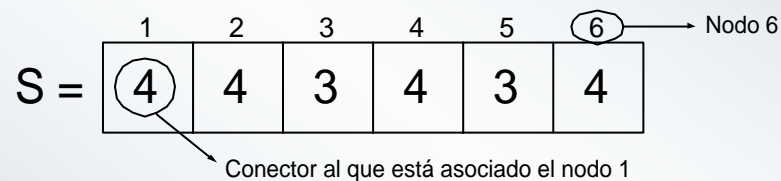
---

1. **Generación de la población inicial**: aleatoria
2. **Representación**: vector que almacena la DMU eficiente sobre la que se proyecta cada DMU no eficiente
3. **Selección**: torneo binario
4. **Enfoques (2 variantes)**: generacional con elitismo (1 individuo) / estacionario
5. **Mutación**: elegir una DMU no eficiente y cambiar su proyección por otra DMU eficiente que la domine
6. **Cruce**: Se escogen dos puntos de corte que determinan tres subcadenas. Cada hijo contiene la subcadena central de uno de los padres y las otras dos del otro padre

## 9. APLICACIONES

### 9.3. *p*-Hub Medio

1. **Generación de la población inicial:** aleatoria
2. **Representación:** vector de enteros de tamaño igual al número de nodos donde el contenido de cada casilla indica el conector asignado al índice correspondiente



3. **Selección:** torneo binario
4. **Enfoques (2 variantes):** generacional con elitismo (1 individuo) / estacionario
5. **Mutación:** el operador de vecino ya conocido (con 0,4 de probabilidad de realizar un cambio de asignación y 0,6 de realizar un cambio de localización)

# 9. APLICACIONES

## 9.3. *p*-Hub Medio

6. **Cruce:** Se compone de los siguientes pasos:
- a) **Localización:** Si algún conector coincide en los dos padres se incluye directamente en el hijo. El resto de conectores se decide aleatoriamente entre los contenidos por uno de los padres
  - b) **Asignación:** Una vez definidos los conectores, se revisa uno a uno los nodos no conectores. Pueden suceder tres casos:
    - El nodo está asignado al mismo conector en ambos padres y, por tanto, ese conector fue incluido en el hijo. En este caso, se asigna directamente a dicho conector
    - El nodo está asignado a un conector distinto en cada padre y uno de ellos coincide con uno de los conectores elegidos para el nuevo hijo. En este caso se asigna directamente a dicho conector
    - El nodo está asignado a un conector distinto en cada padre y ninguno de ellos coincide con alguno de los conectores elegidos para el nuevo hijo. En este caso se asigna aleatoriamente a uno de los nuevos conectores

## 9. APLICACIONES

### 9.3. *p*-Hub Medio. Ejemplos de Cruce

$$\text{Padre}_1 = (3 \ 4 \ \underline{3} \ \underline{4} \ 3)$$

$$\text{Padre}_2 = (3 \ 3 \ \underline{3} \ \underline{4} \ 4)$$

#### Localización:

Como los dos conectores coinciden en ambos padres, se consideran en el hijo

$$\text{Hijo} = (* \ * \ \underline{3} \ \underline{4} \ *)$$

#### Asignación:

El nodo 1 se asigna al mismo conector en ambos padres, así que se realiza esa asignación en el hijo

$$\text{Hijo} = (3 \ * \ \underline{3} \ \underline{4} \ *)$$

El resto de asignaciones se deciden aleatoriamente. P.ej.:

$$\text{Hijo} = (3 \ 3 \ \underline{3} \ \underline{4} \ 3)$$

## 9. APLICACIONES

### 9.3. *p*-Hub Medio. Ejemplos de Cruce

$$\text{Padre}_1 = (3 \ 3 \ \underline{3} \ \underline{4} \ 4)$$

$$\text{Padre}_2 = (\underline{1} \ 1 \ \underline{3} \ 3 \ 3)$$

#### Localización:

El conector 3 coincide en ambos padres

$$\text{Hijo} = (* \ * \ \underline{3} \ * \ *)$$

Se elige aleatoriamente un conector entre 1 ó 4. Por ejemplo, 1

$$\text{Hijo} = (\underline{1} \ * \ \underline{3} \ * \ *)$$

#### Asignación:

Los nodos 4 y 5 se asignan directamente al conector 3

$$\text{Hijo} = (\underline{1} \ * \ \underline{3} \ 3 \ 3)$$

El nodo 2 se asigna aleatoriamente a 1 ó 3. Por ejemplo, 1

$$\text{Hijo} = (\underline{1} \ 1 \ \underline{3} \ 3 \ 3)$$



## 9. APLICACIONES

### 9.3. $p$ -Hub Medio. Ejemplos de Cruce

$$\text{Padre}_1 = (3 \ 3 \ \underline{3} \ \underline{4} \ 4)$$

$$\text{Padre}_2 = (\underline{1} \ \underline{2} \ 1 \ 1 \ 2)$$

#### Localización:

Ningún conector coincide, así que se seleccionan dos aleatoriamente entre 1, 2, 3 ó 4. Por ejemplo, 2 y 3

$$\text{Hijo} = (* \ \underline{2} \ \underline{3} \ * \ *)$$

#### Asignación:

Los nodos 1 y 5 se asignan directamente a los conectores 3 y 2, respectivamente

$$\text{Hijo} = (3 \ \underline{2} \ \underline{3} \ * \ 2)$$

El nodo 4 se asigna aleatoriamente a 2 ó 3. Por ejemplo, 2

$$\text{Hijo} = (3 \ \underline{2} \ \underline{3} \ 2 \ 2)$$



# 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*

➤ ***SON ATRACTIVOS DESDE UN PUNTO  
DE VISTA COMPUTACIONAL***

# BIBLIOGRAFÍA

---

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

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

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

# ALGORÍTMICA

## 2003 - 2004

- 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. Modelos Híbridos I: Algoritmos Meméticos
  - Tema 10. Modelos Híbridos II: *Scatter Search*
  - Tema 11. Modelos Híbridos III: Otras Hibridaciones
- Parte VI. Conclusiones
  - Tema 12. Algunas Consideraciones sobre la Adaptación de Metaheurísticas a la Resolución de Problemas