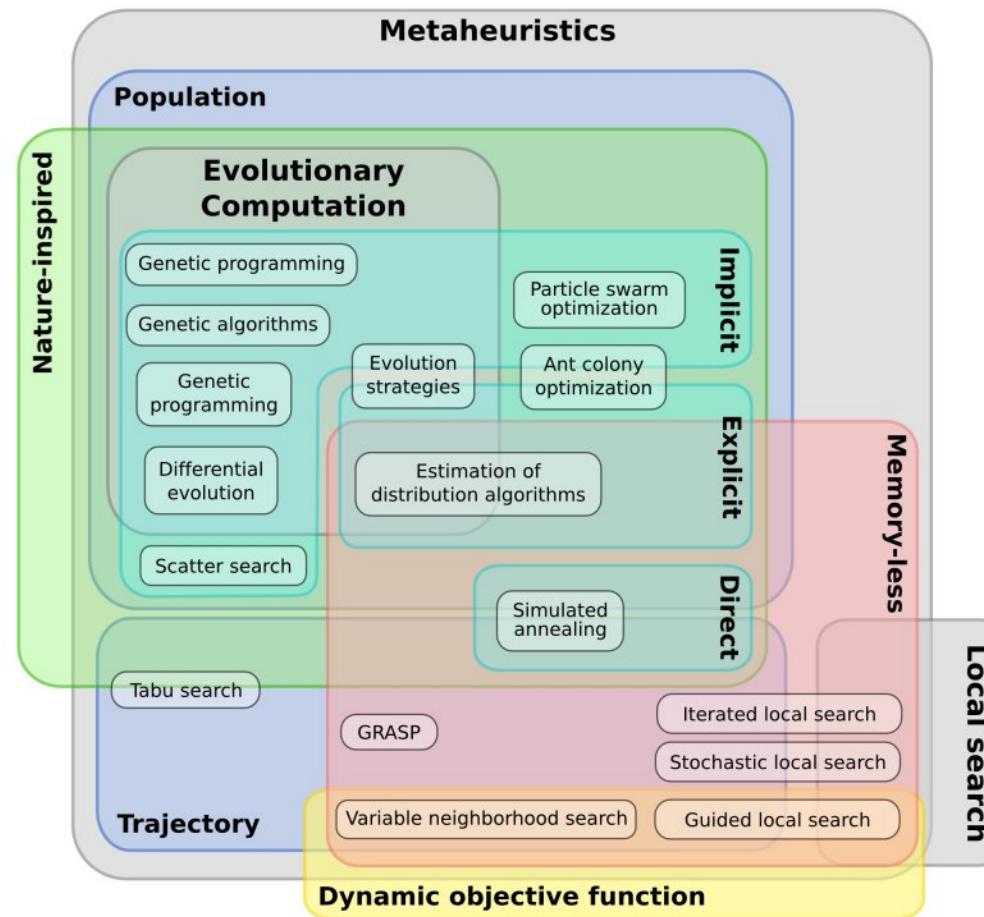


# Tema 3. Metaheurísticas basadas en poblaciones. Algoritmos evolutivos

Profesor: Sebastián Ventura



# Objetivos

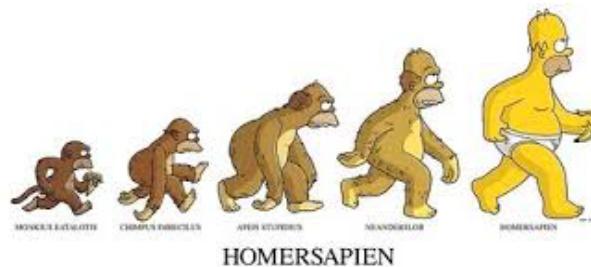
- Conocer las características de la computación evolutiva
- Entender el funcionamiento y estructura general de los algoritmos evolutivos
- Conocer los diferentes tipos de metaheurísticas basadas en población

# Índice

- **Introducción a la computación evolutiva**
- Algoritmos genéticos
- Diversidad Vs Convergencia

# Introducción a la computación evolutiva

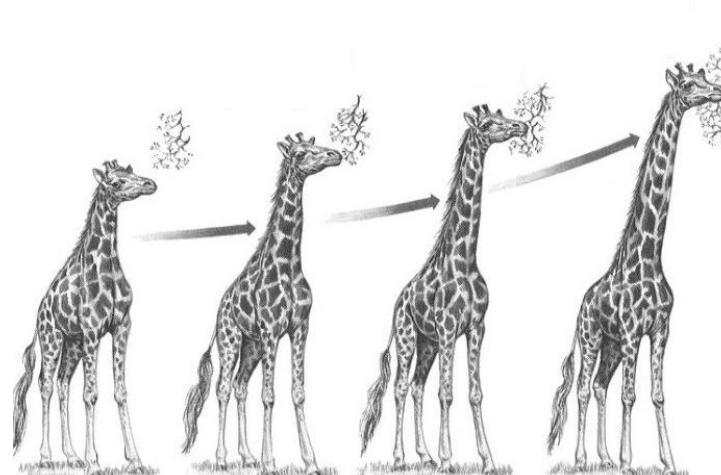
- En la naturaleza, los **procesos evolutivos** ocurren cuando se dan las **siguientes condiciones**:
  - Un individuo tiene la **habilidad de reproducirse**
  - **Existe una población** de individuos que pueden reproducirse
  - Existe alguna variedad o **habilidad entre los individuos** que se reproducen
  - Algunos individuos **sobreviven en el entorno** gracias a esa habilidad
  - El proceso **no tiene memoria**



S. Ventura - J. M. Luna

# Introducción a la computación evolutiva

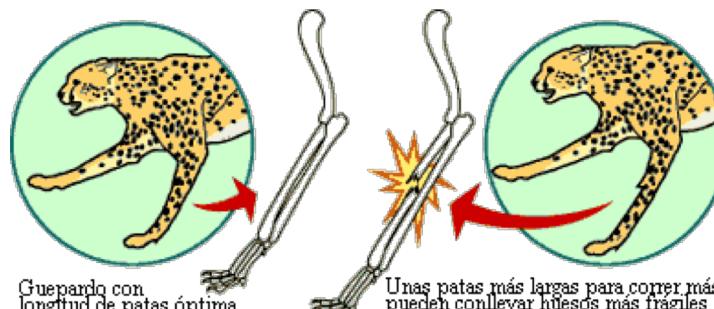
- Las **jirafas primitivas** tenían el cuello corto y las patas de longitud normal. En épocas de sequía, las jirafas intentaba alcanzar las hojas de las ramas más altas. El cuello y patas de las nuevas jirafas eran cada vez más largos y estas variaciones se transmitían hereditariamente



S. Ventura - J. M. Luna

# Introducción a la computación evolutiva

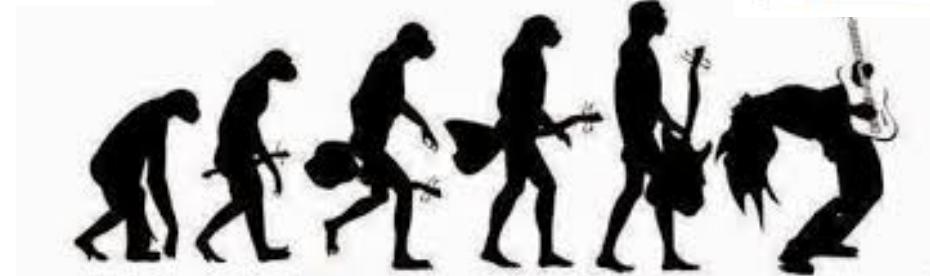
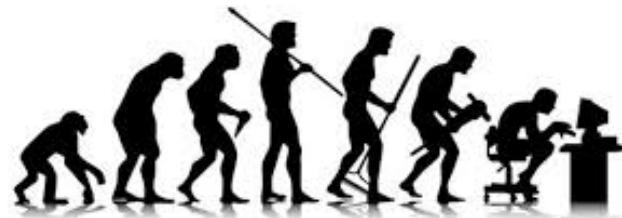
- Para **evolucionar** hacia habilidades mejores es necesario que exista variabilidad genética. Un guepardo podría correr más rápido sólo si hay suficientes “*genes para correr más rápido*” en la población
- Debe **existir una compensación**, pues algunas características mejores hacen que otras empeoren. Extremidades más largas pueden permitir al guepardo dar una mayor zancada. Sin embargo, las patas serían más débiles y fallarían



S. Ventura - J. M. Luna

# Introducción a la computación evolutiva

- En el proceso de evolución, el entorno (medio ambiente) juega un papel fundamental y los **individuos** se van **adaptando a dicho entorno**



# Introducción a la computación evolutiva

- **Computación evolutiva:** simulación del proceso evolutivo en un ordenador. Esta técnica de optimización probabilística mejora, con cierta frecuencia, a otros métodos clásicos en problemas difíciles
- La **computación evolutiva** está compuesta por modelos de evolución basados en poblaciones cuyos individuos representan soluciones a problemas
- Basada en **principios darwinianos**: reproducción y selección natural

# Introducción a la computación evolutiva

- Existen **cuatro paradigmas** básicos:
  - **Algoritmos Genéticos.** Utilizan operadores genéticos sobre cromosomas.
  - **Estrategias de Evolución.** Enfatizan los cambios de comportamiento al nivel de los individuos.
  - **Programación Evolutiva.** Enfatizan los cambios de comportamiento al nivel de las especies.
  - **Programación Genética.** Evoluciona expresiones que representan programas.
- Otros modelos de evolución de poblaciones:
  - **EDA.** Basados en estimación de distribuciones
  - **DE.** Evolución diferencial
  - **Algoritmos meméticos**

# Índice

- Introducción a la computación evolutiva
- **Algoritmos genéticos**
- Diversidad Vs Convergencia

# Algoritmos genéticos

- Algoritmos de optimización, búsqueda y aprendizaje inspirados en los procesos de evolución natural y evolución genética
- Utilizan los siguientes **mecanismos**:
  - Sobreviven los organismos con mejor capacidad dentro de una población
  - Secuencias de caracteres para representar el ADN
  - Métodos aleatorios para generar la población y su reproducción

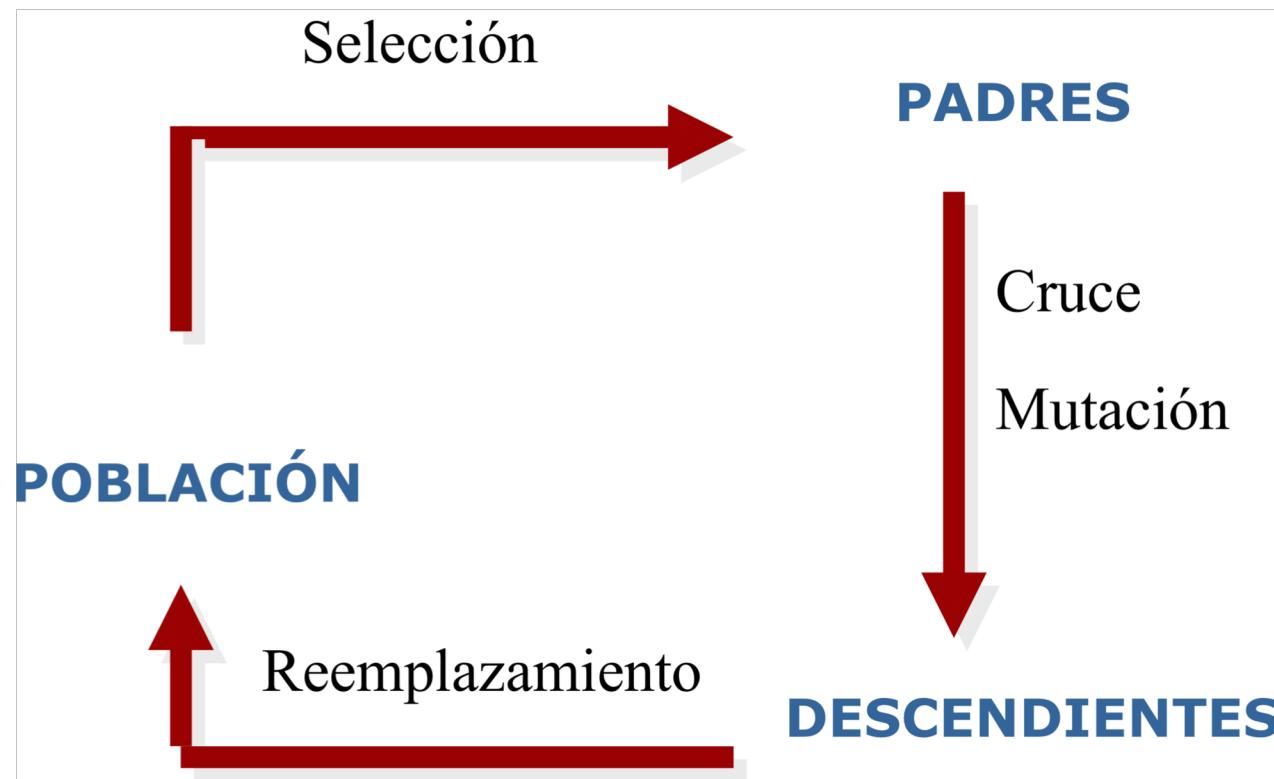


S. Ventura - J. M. Luna

# Algoritmos genéticos

- **Condiciones necesarias para la evolución:**
  - Entidad capaz de reproducirse
  - Población de dichas entidades
  - Variedad en la reproducción
  - Diferencias en la **habilidad de las entidades que permiten sobrevivir** en base al medio ambiente
    - Individuos con mayor éxito tienen mayor probabilidad de reproducirse y de generar un mayor número de descendientes
    - Genes de individuos mejor adaptados se propagarán en sucesivas generaciones

# Algoritmos genéticos



# Algoritmos genéticos

**Inicio**

$t = 0;$

Iniciar P( $t$ );

Evaluar P( $t$ );

**mientras** (no se cumpla la condición de parada) **hacer**

    Seleccionar P( $t+1$ ) desde P( $t$ )

    Cruzar P( $t+1$ )

    Mutar P( $t+1$ )

    Evaluar P( $t+1$ )

    Reemplazar P( $t$ ) por P( $t+1$ )

$t=t+1$

**fin mientras**

**fin**

# Algoritmos genéticos

- **Construir un algoritmo genético:**

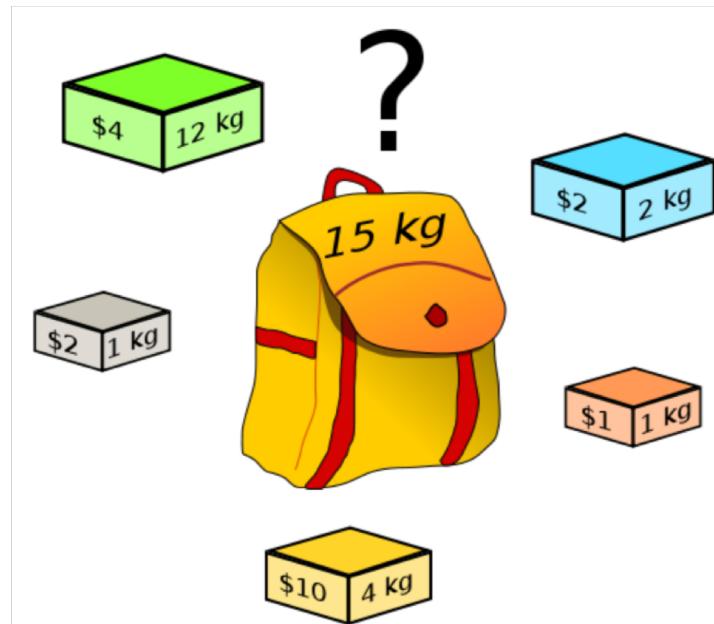
- Establecer un tipo de representación
- Decidir cómo inicializar la población
- Diseñar una correspondencia entre genotipo y fenotipo
- Definir una forma de evaluar individuos
- Diseñar un operador de mutación adecuado
- Diseñar un operador de cruce adecuado
- Decidir un operador de selección de padres
- Establecer un reemplazo de individuos
- Definir una condición de parada

*¿Ideas para cada punto?*

# Algoritmos genéticos

- Construir un algoritmo genético:

*¿Ideas para cada punto?*



# Algoritmos genéticos

- **Construir un algoritmo genético:**
  - Movimiento: arriba, abajo, derecha, izquierda
  - Evaluación: camino de 5 pasos con menor costo
  - Definir el resto:
    - Representación
    - Cruce y mutación
    - Selección y reemplazo
    - Parada

*¿Ideas para cada punto?*

5	4	2	2	4
4	3	2	1	1
5	1	3	0	6
4	1	2	5	7
2	1	3	5	4

# Índice

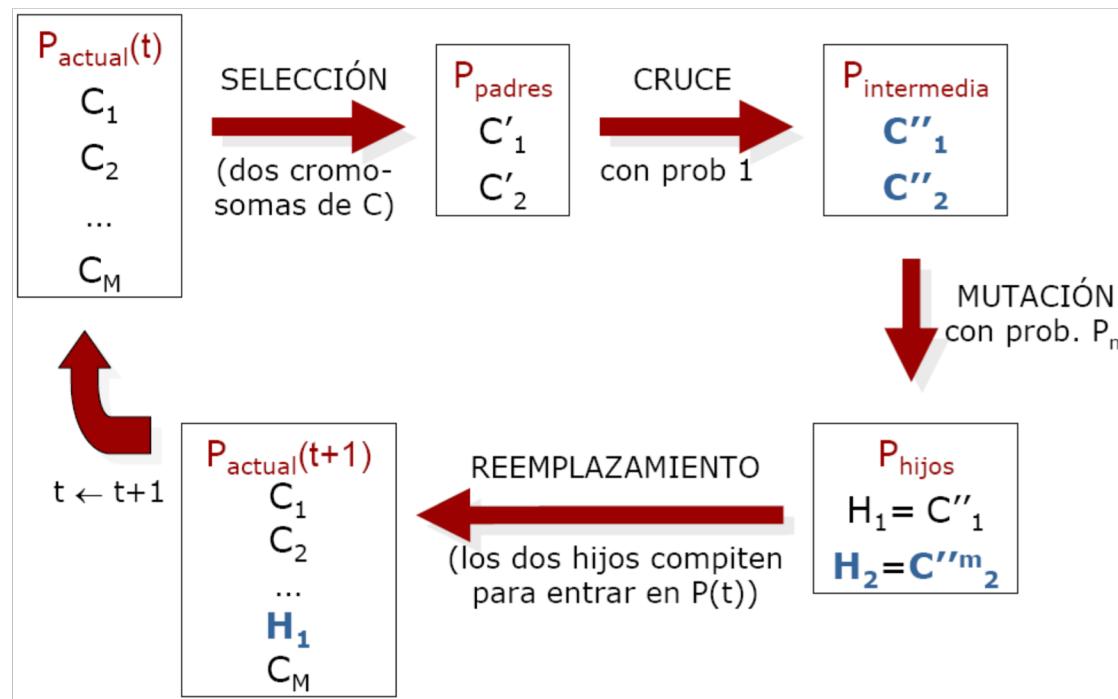
- Introducción a la computación evolutiva
- **Algoritmos genéticos**
  - Modelos

# Algoritmos genéticos: Modelos

- **Modelo generacional:** En cada generación se crea una población completa con nuevos individuos.
  - Puede o no llevar elitismo (el mejor individuo es mantenido)
- **Modelo estacionario:** En cada generación se escogen dos o más padres de la población y se les aplica operadores genéticos. Los nuevos individuos compiten por entrar en la población
  - Produce una **convergencia rápida** cuando se **reemplazan los peores cromosomas de la población**

# Algoritmos genéticos: Modelos

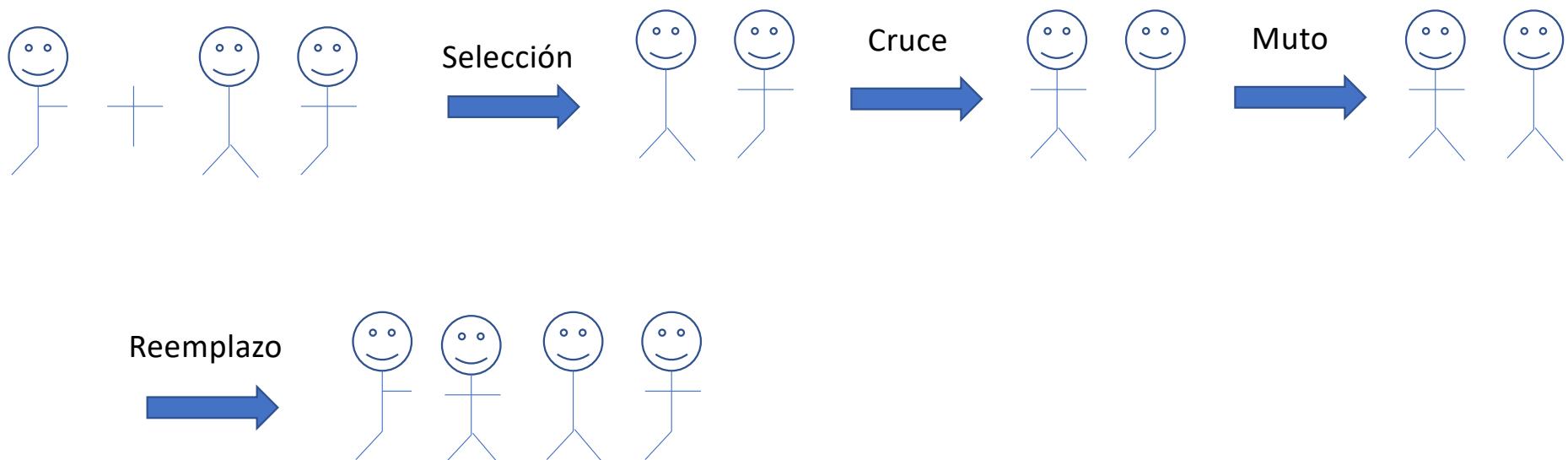
- **Modelo estacionario**



S. Ventura - J. M. Luna

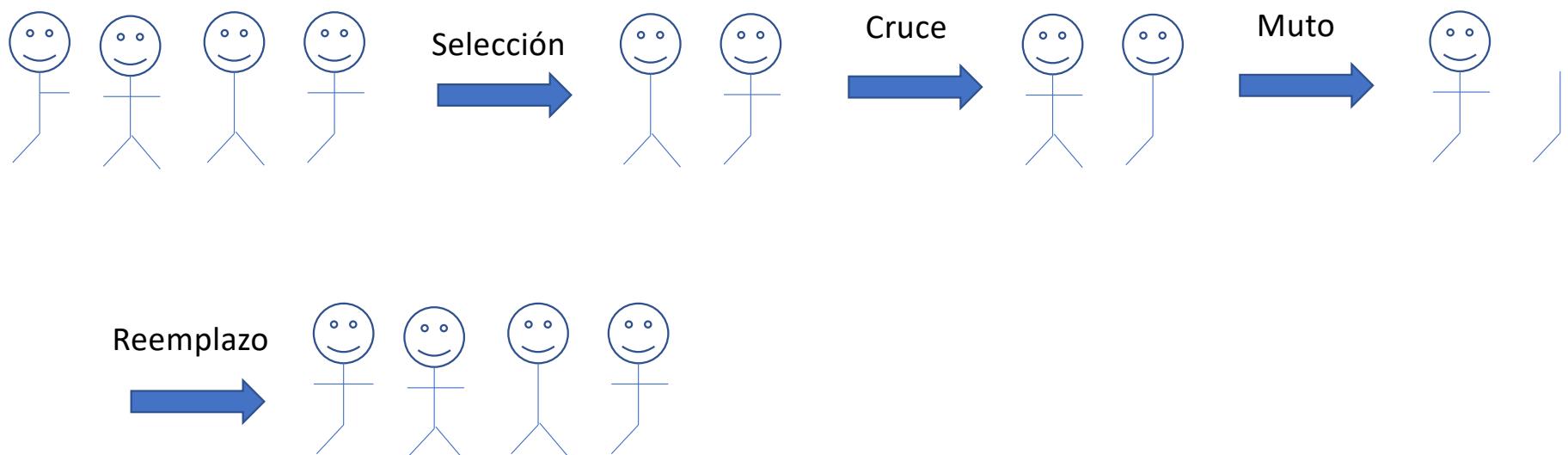
# Algoritmos genéticos: Modelos

- **Modelo estacionario. Generación 0**



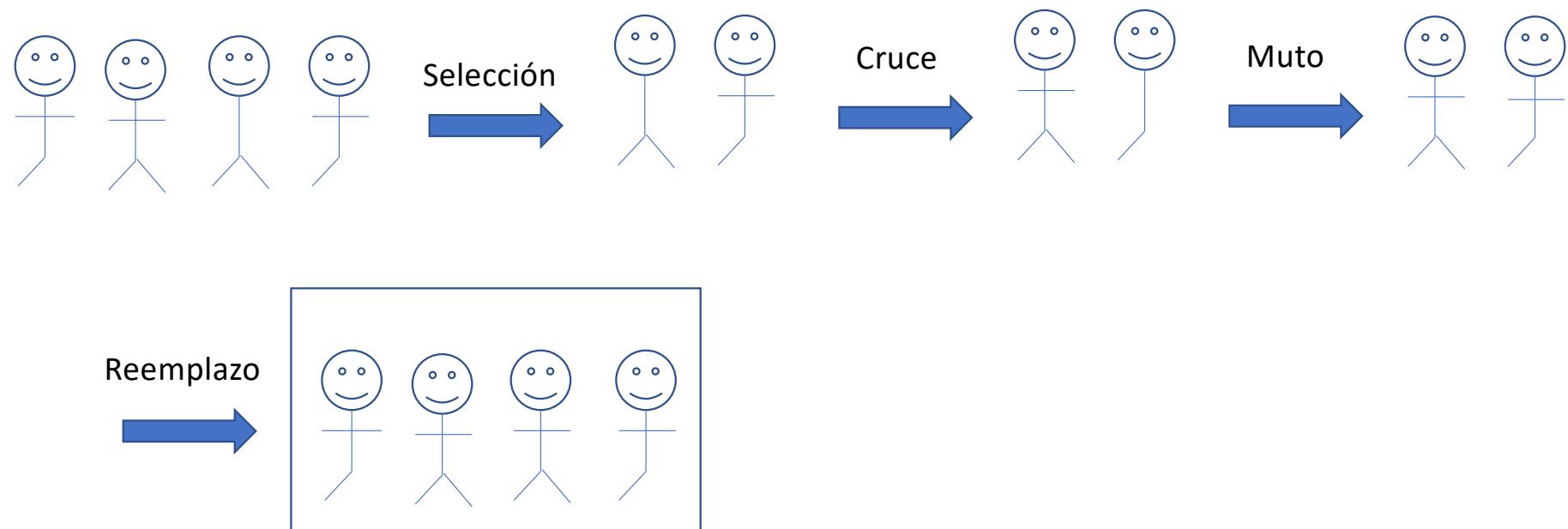
# Algoritmos genéticos: Modelos

- **Modelo estacionario. Generación 1**



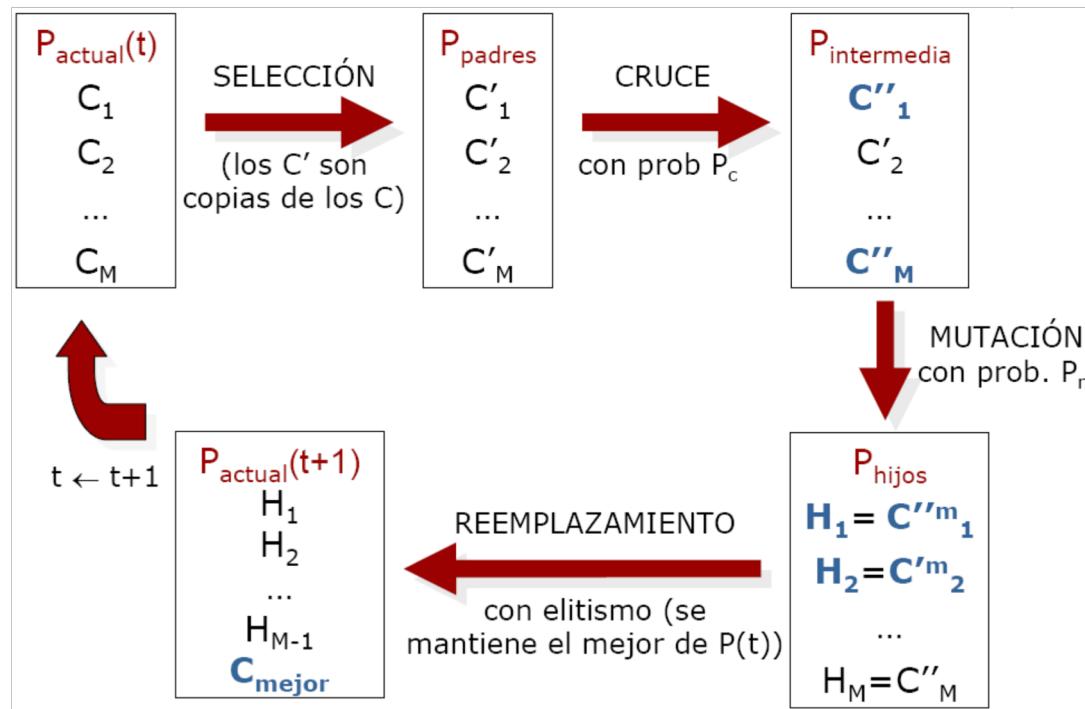
# Algoritmos genéticos: Modelos

- **Modelo estacionario. Generación 2**



# Algoritmos genéticos: Modelos

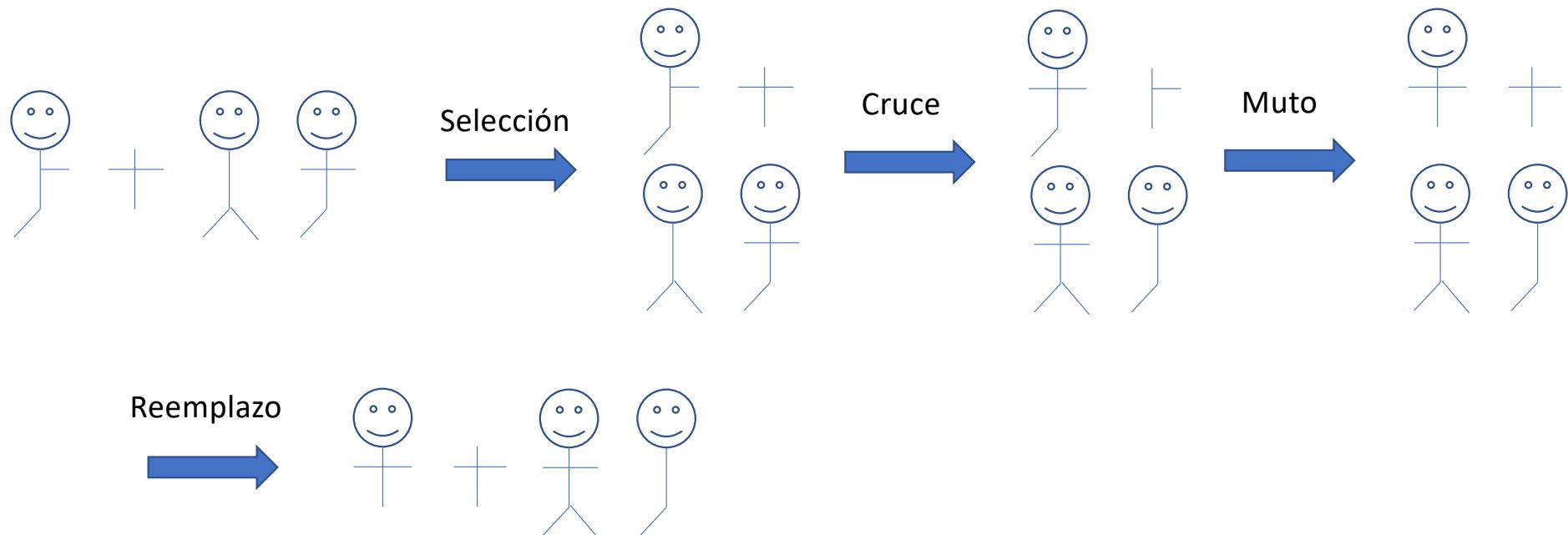
- **Modelo generacional**



S. Ventura - J. M. Luna

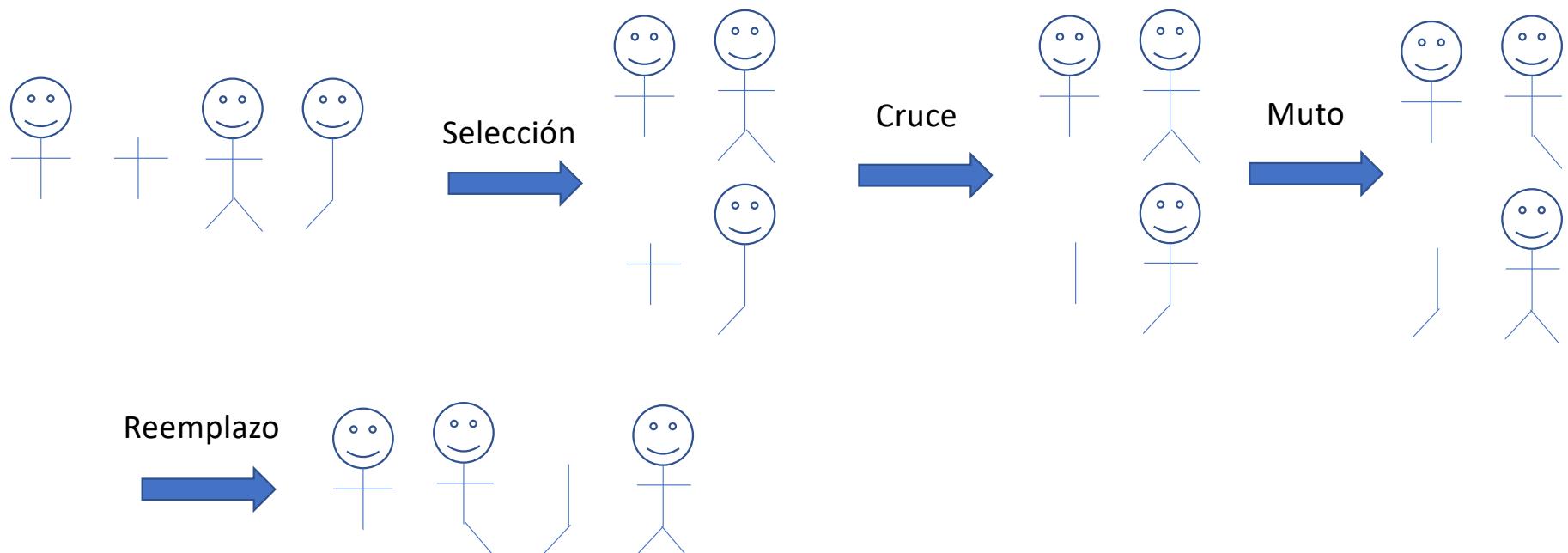
# Algoritmos genéticos: Modelos

- **Modelo generacional. Generación 0**



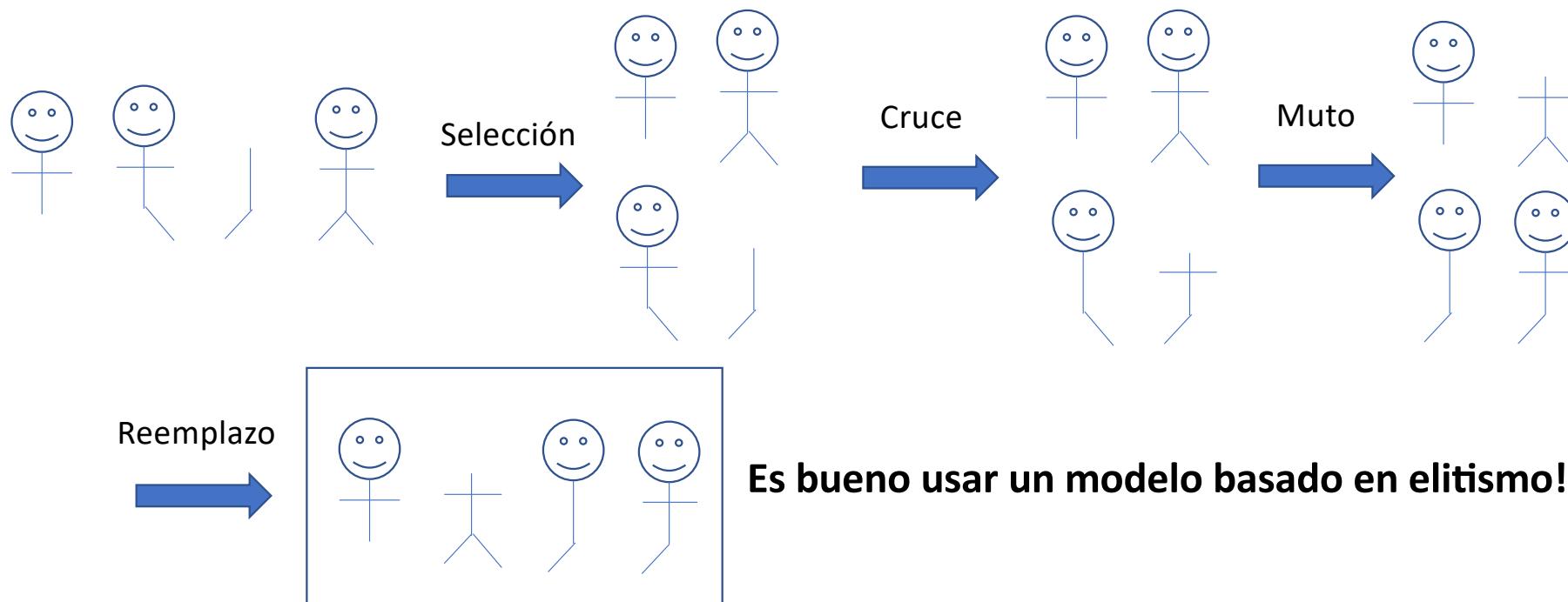
# Algoritmos genéticos: Modelos

- **Modelo generacional. Generación 1**



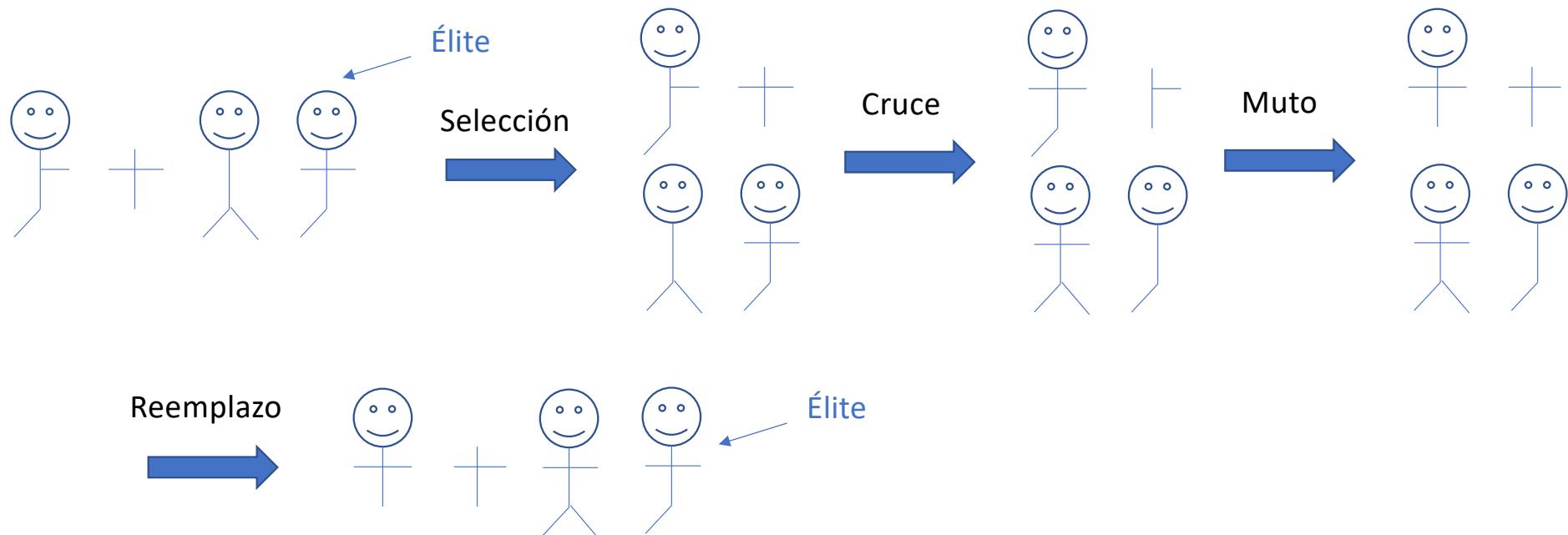
# Algoritmos genéticos: Modelos

- **Modelo generacional. Generación 2**



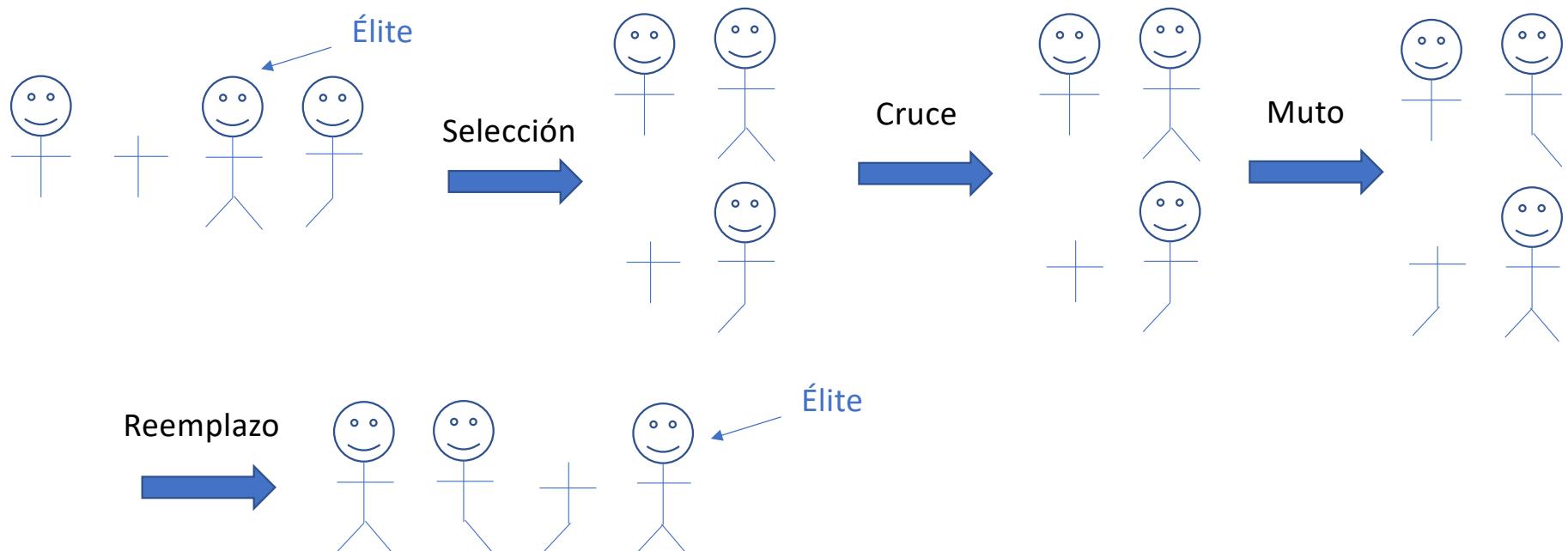
# Algoritmos genéticos: Modelos

- **Modelo generacional elitista. Generación 0**



# Algoritmos genéticos: Modelos

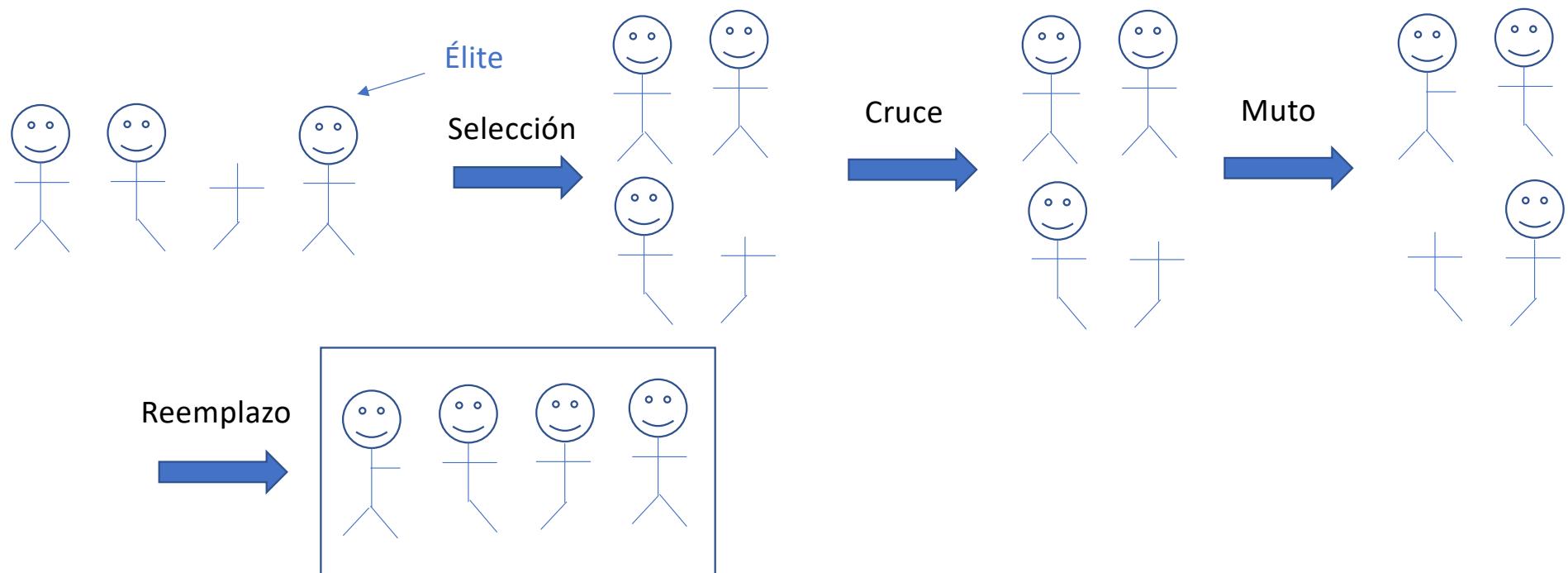
- **Modelo generacional. Generación 1**



S. Ventura - J. M. Luna

# Algoritmos genéticos: Modelos

- **Modelo generacional. Generación 2**



S. Ventura - J. M. Luna

# Índice

- Introducción a la computación evolutiva
- **Algoritmos genéticos**
  - Modelos
  - **Primeros pasos en la construcción de un AG**

# Algoritmos genéticos: Primeros pasos

- **Representación:** debemos disponer de un mecanismo para codificar un individuo como un genotipo. Los algoritmos genéticos han sido generalmente representados con codificación binaria
  - Tipos de representaciones:
    - Binaria
    - Ordinal
    - Entera
    - Real
  - Importancia de diferenciar entre Genotipo y Fenotipo

# Algoritmos genéticos: Primeros pasos

- **Inicialización:** debemos establecer claramente cómo vamos a inicializar la población
  - Distribución uniforme sobre el espacio de búsqueda
    - Binaria: 0 ó 1 con probabilidad 0.5
    - Real: uniforme sobre un intervalo dado
  - Elegir una población a partir de los resultados de una heurística previa

# Algoritmos genéticos: Primeros pasos

- **Evaluación:** representación de la aptitud de cada uno de los individuos.
  - Es el paso más costoso en cualquier algoritmo evolutivo
  - Puede ser una subrutina, una simulación o cualquier proceso externo
  - Se pueden utilizar funciones aproximadas para reducir el costo de evaluación
  - Se pueden introducir penalizaciones cuando hay restricciones

# Índice

- Introducción a la computación evolutiva
- **Algoritmos genéticos**
  - Modelos
  - Primeros pasos en la construcción de un AG
  - **Estrategias de selección**

# Algoritmos genéticos: Estrategias de selección

- Debemos garantizar que los **mejores individuos tienen una mayor posibilidad de reproducirse** frente a los individuos menos buenos
- Dar una oportunidad de reproducirse a los individuos menos buenos
  - Pueden incluir material genético útil
- **Presión selectiva:** en qué grado la reproducción está dirigida por los mejores individuos

# Algoritmos genéticos: Estrategias de selección

- **Estrategias de selección** más comunes:

- Selección aleatoria
- Selección por torneo
- Selección por ruleta
- Selección basada en ranking
- Emparejamiento variado inverso

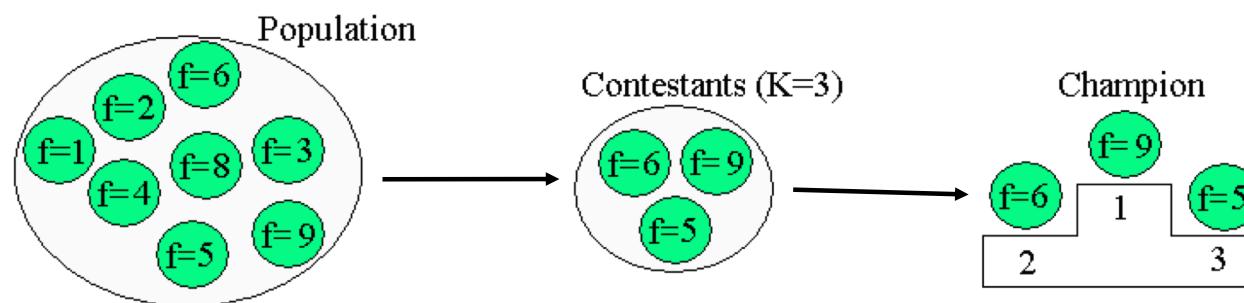
# Algoritmos genéticos: Estrategias de selección

- **Estrategias de selección** más comunes:
  - **Selección aleatoria:** Se escogen padres de manera aleatoria de entre la población:
    - Con reemplazamiento
    - Sin reemplazamiento

# Algoritmos genéticos: Estrategias de selección

- **Estrategias de selección** más comunes:

- **Selección por torneo:** para cada parent a seleccionar
  - Se escoge aleatoriamente  $k$  individuos (con reemplazamiento)
  - Se selecciona el mejor de ellos
  - $k$  es el tamaño del torneo. A mayor  $k$  mayor presión selectiva y viceversa
  - Con  $k=1$  es equivalente a una selección aleatoria



# Algoritmos genéticos: Estrategias de selección

- **Estrategias de selección** más comunes:
  - **Selección por torneo:** presión selectiva
    - **Torneos grandes:** hay una presión selectiva elevada y los peores individuos apenas tienen oportunidades de reproducirse
      - **Caso particular:** elitismo global, donde participan todos los individuos y la selección se vuelve totalmente determinista
    - **Torneos pequeños:** la presión selectiva disminuye y los peores individuos tienen más oportunidades de ser seleccionados

# Algoritmos genéticos: Estrategias de selección

- **Estrategias de selección** más comunes:

- **Selección por ruleta:** también conocida como selección proporcional a la función de aptitud. Es uno de los métodos más utilizados.

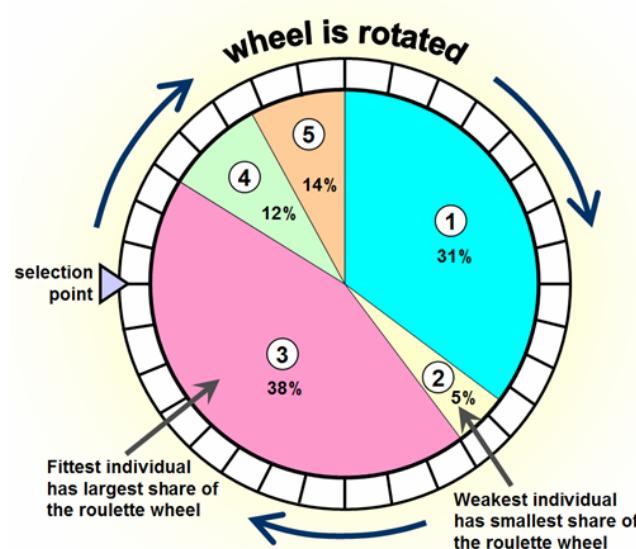
- A cada individuo se le asigna una parte proporcional a su función de aptitud de una ruleta
- La suma de todos los porcentajes es igual a la unidad
- Los mejores individuos tendrán una proporción de ruleta mayor que los peores
- Dados N individuos, la probabilidad asociada a su selección es:

$$p_i = \frac{f_i}{\sum_{j=1}^N f_j}$$

- Generalmente la población está ordenada en base a la función de aptitud por lo que las proporciones más grandes están al inicio de la ruleta
- La selección se realiza con un aleatorio en el intervalo [0, 1] y devuelve el individuo en esa posición de la ruleta

# Algoritmos genéticos: Estrategias de selección

- **Estrategias de selección** más comunes:
  - Selección por ruleta:



S. Ventura - J. M. Luna

# Algoritmos genéticos: Estrategias de selección

- **Estrategias de selección** más comunes:

- **Selección basada en ranking:** la población se ordena de acuerdo a su función de aptitud y las probabilidades de selección se asignan en base a su ranking
  - Puede evitar una convergencia prematura
  - Puede ser computacionalmente costosa (ordenar la población)
  - El individuo peor tiene la posición 1 y el mejor la posición N
  - Se aplica una selección por ruleta normal, pero aplicada a los rankings

# Algoritmos genéticos: Estrategias de selección

- **Estrategias de selección** más comunes:

- **Emparejamiento variado inverso:**

- Se selecciona aleatoriamente un padre
    - Para el otro parente, se seleccionan N padres y se escoge el más lejano al primero
    - Está orientado a generar diversidad

# Algoritmos genéticos: Estrategias de selección

- **Comparativa selección por ruleta Vs basada en ranking**

- Dados 5 individuos con los valores de fitness (maximizar):
  - 0.97, 0.55, 0.43, 0.14, 0.01

- **Ruleta**

- $0.97/2.10=0.46$
- $0.55/2.10=0.26$
- $0.43/2.10=0.20$
- $0.14/2.10=0.07$
- $0.01/2.10=0.01$

- **Basado en ranking**

- $5/15=0.33$
- $4/15=0.27$
- $3/15=0.20$
- $2/15=0.13$
- $1/15=0.07$

# Algoritmos genéticos

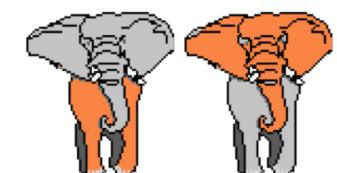
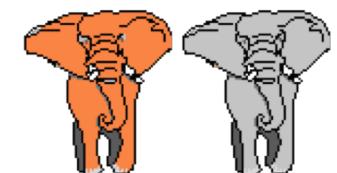


# Índice

- Introducción a la computación evolutiva
- **Algoritmos genéticos**
  - Modelos
  - Primeros pasos en la construcción de un AG
  - Estrategias de selección
  - **Operadores de cruce**

# Algoritmos genéticos: Operadores de cruce

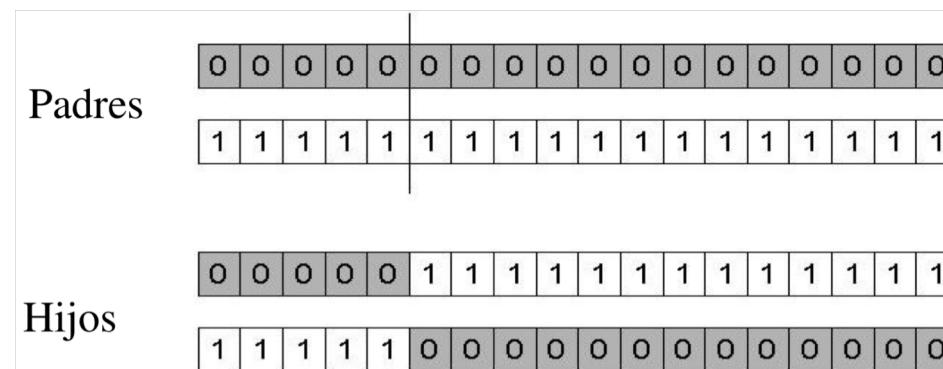
- Podría tenerse uno o más operadores de cruce para una representación
- Algunos aspectos importantes:
  - Los hijos deberán heredar algunas **características de cada padre**
  - Se debe diseñar de acuerdo a la representación
  - Se deben producir cromosomas válidos
- **Operadores de cruce** más comunes:
  - Cruce en 1 punto
  - Cruce en  $n$  puntos
  - Cruce uniforme
  - Cruce aritmético simple
  - Cruce BLX- $\alpha$
  - Cruces sobre representación ordinal (permutaciones)



# Algoritmos genéticos: Operadores de cruce

- **Cruce en 1 punto**

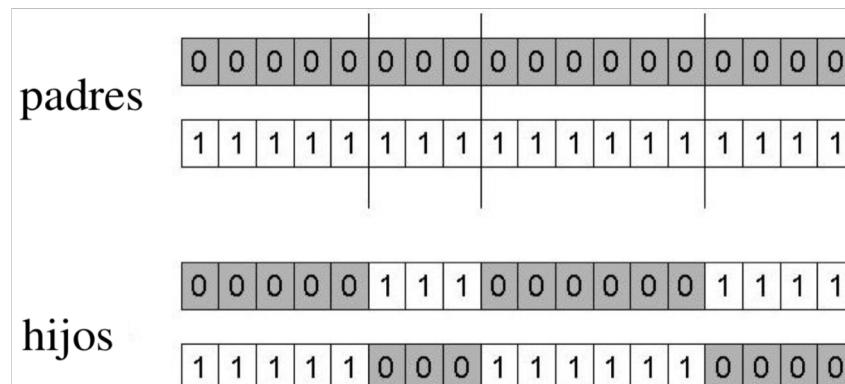
- Seleccionar un punto al azar en ambos padres
  - Cortar los padres en este punto
  - Crear hijos intercambiando las colas
  - Probabilidad de cruce típica en el rango (0.6, 0.9)



# Algoritmos genéticos: Operadores de cruce

- **Cruce en  $n$  puntos**

- Seleccionar  $n$  puntos al azar en ambos padres
- Cortar los padres en este punto
- Pegar las partes alternando entre los padres
- Generalización del cruce en 1 punto

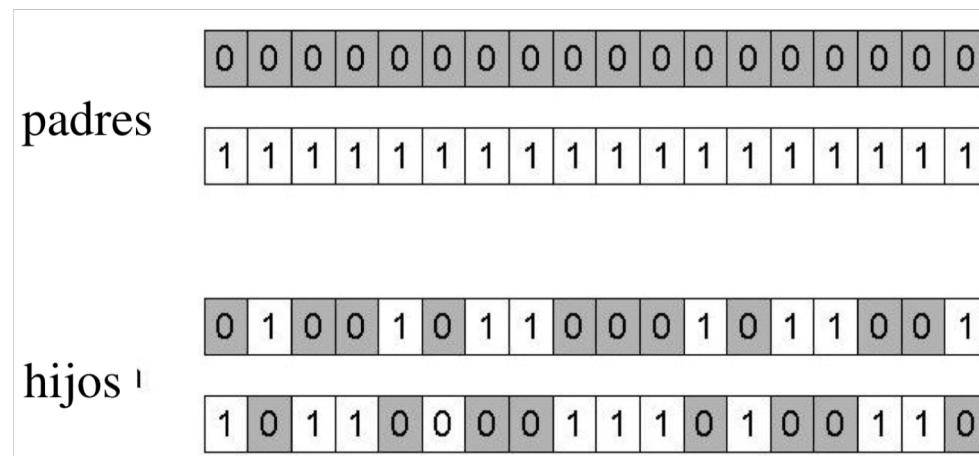


S. Ventura - J. M. Luna

# Algoritmos genéticos: Operadores de cruce

- **Cruce uniforme**

- Lanzar un aleatorio  $[0, 1]$  para cada gen de los padres
  - En base a dicho aleatorio, tomar el gen de uno u otro parente
  - El otro hijo será el inverso



# Algoritmos genéticos: Operadores de cruce

- **Cruce aritmético simple** (representación real)
  - El gen de la descendencia toma el valor medio de los genes de los padres
  - Tiene la desventaja de que únicamente se genera un descendiente

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

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

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

# Algoritmos genéticos: Operadores de cruce

- **Cruce BLX- $\alpha$**  (representación real)
  - Dados 2 cromosomas:  $C_1 = (c_{11}, \dots, c_{1n})$  y  $C_2 = (c_{21}, \dots, c_{2n})$
  - BLX- $\alpha$  genera dos descendientes:  $H_k = (h_{k1}, \dots, h_{ki}, \dots, h_{kn})$ ,  $k = 1, 2$
  - $h_{ki}$  se genera aleatoriamente en el intervalo:  $[C_{\min} - l \cdot \alpha, C_{\max} + l \cdot \alpha]$

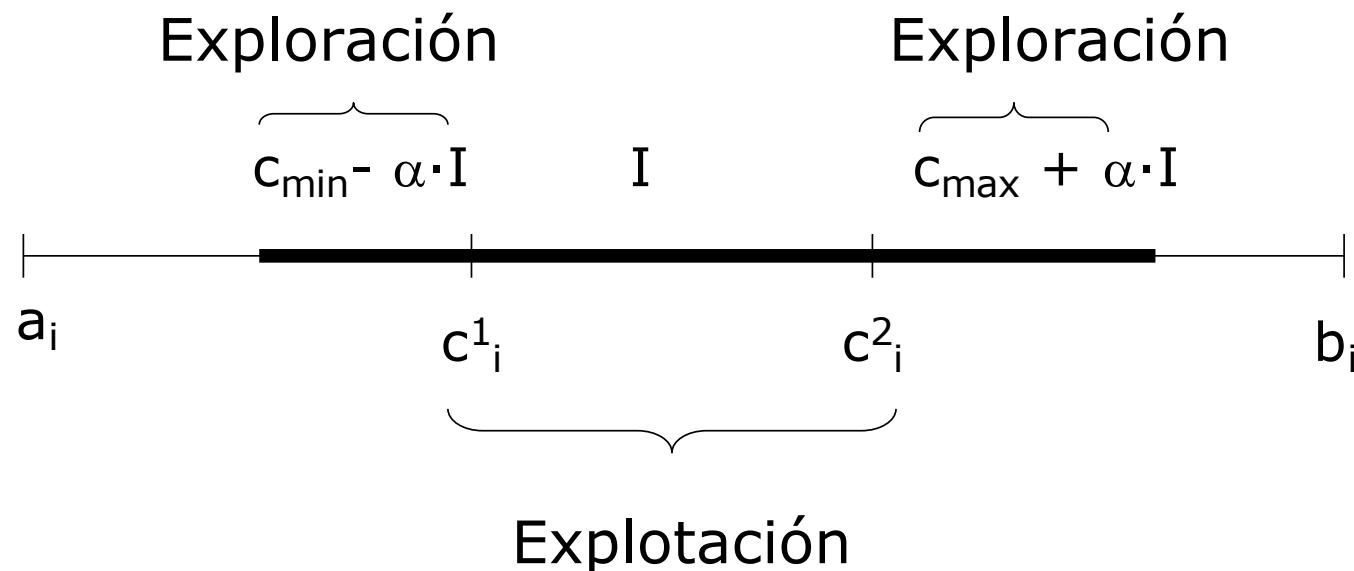
$$C_{\max} = \max \{c_{1i}, c_{2i}\}$$

$$C_{\min} = \min \{c_{1i}, c_{2i}\}$$

$$l = C_{\max} - C_{\min}, \alpha \in [0, 1]$$

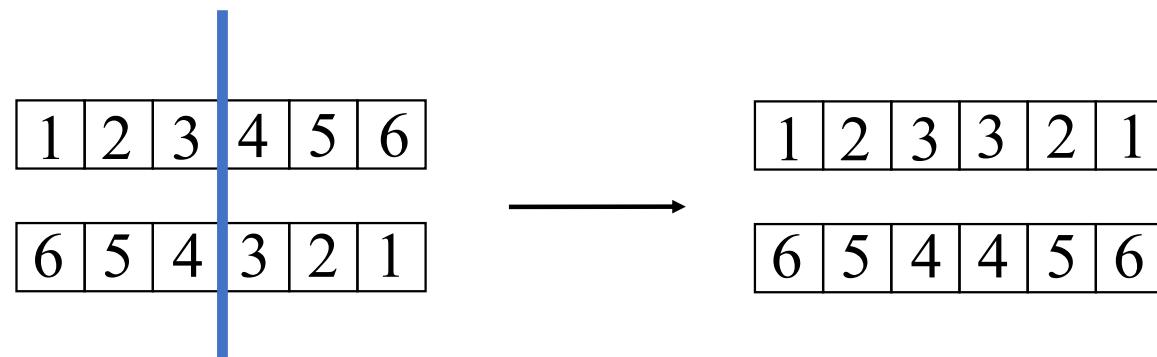
# Algoritmos genéticos: Operadores de cruce

- **Cruce BLX- $\alpha$**  (representación real)



# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
  - Los cruces estándar generalmente conlleva a soluciones inadmisibles



# Algoritmos genéticos: Operadores de cruce

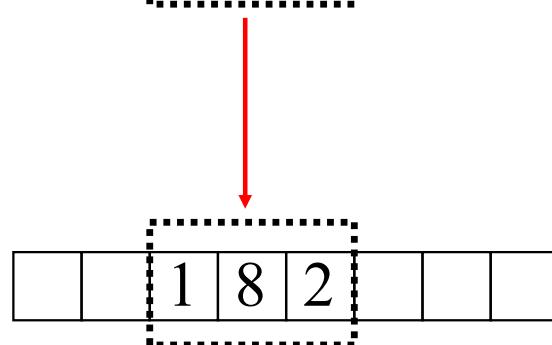
- **Cruces para permutaciones** (representación ordinal)
  - Operador de cruce OX1

Padre 1

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

Padre 2

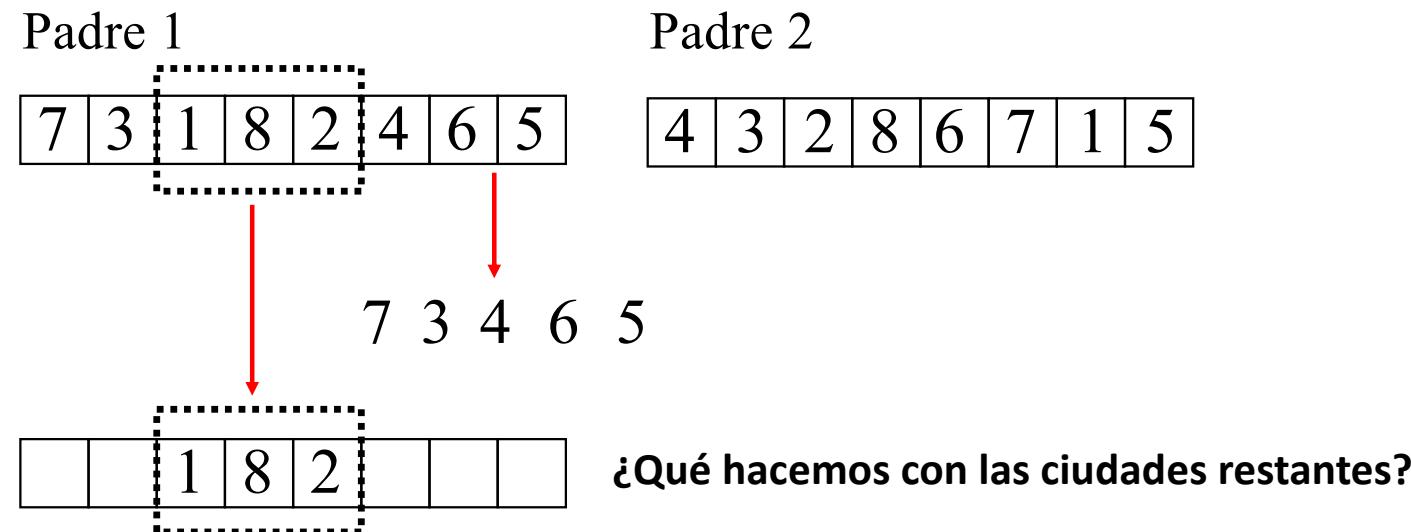
4	3	2	8	6	7	1	5
---	---	---	---	---	---	---	---



¿Qué hacemos con las ciudades restantes?

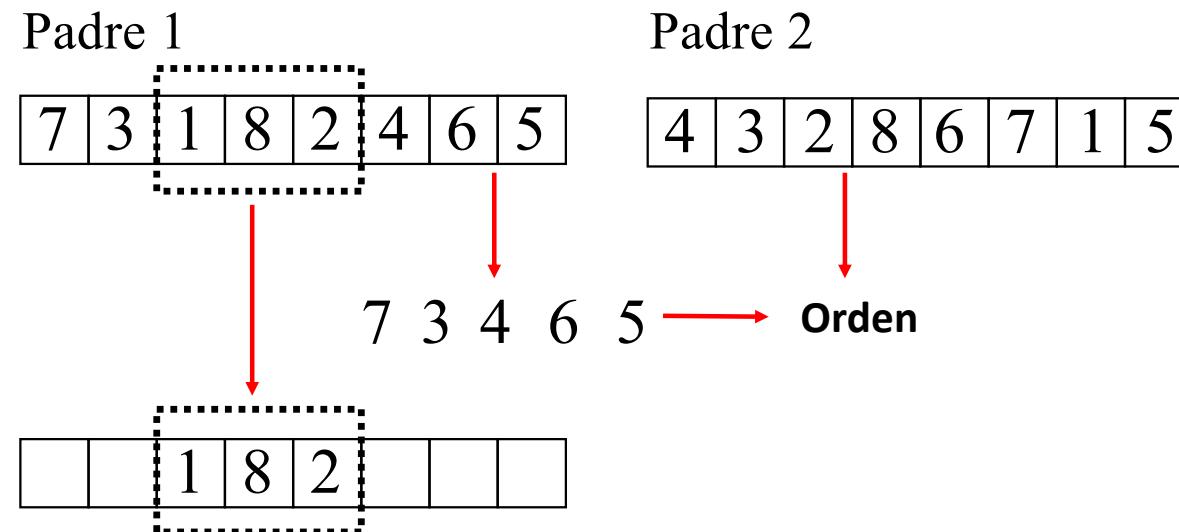
# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
  - Operador de cruce OX1



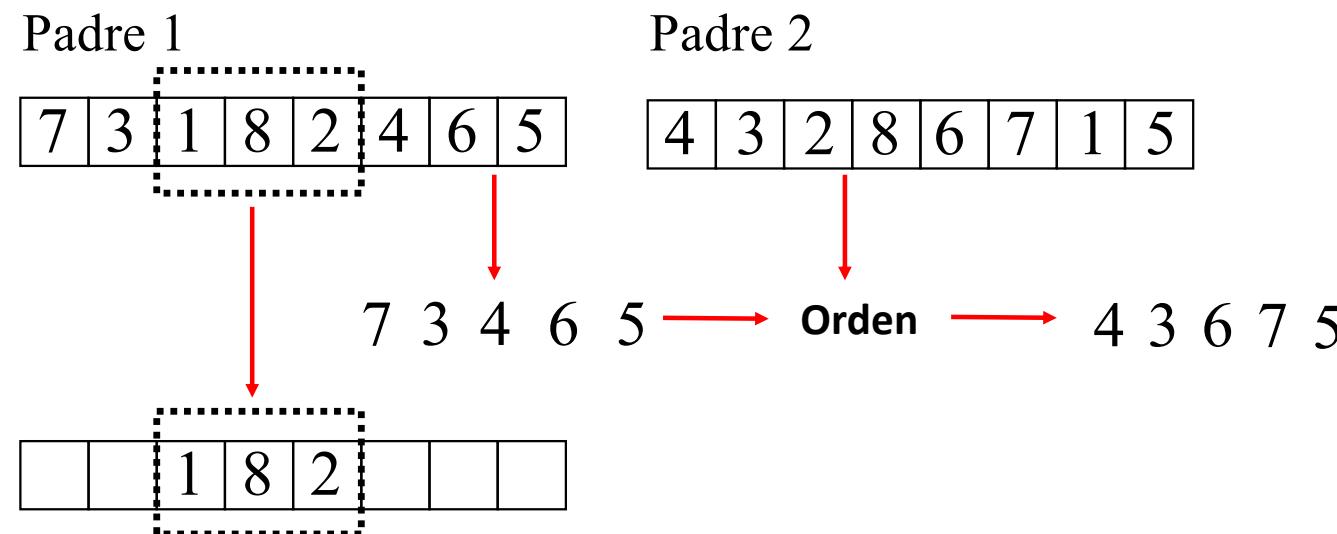
# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
- Operador de cruce OX1



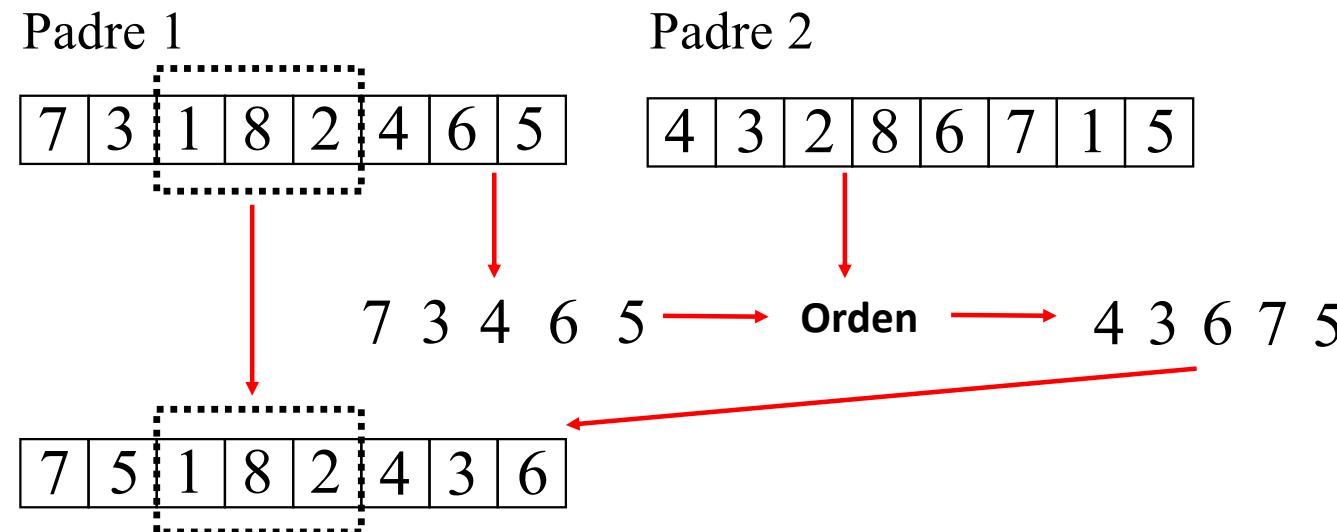
# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
    - Operador de cruce OX1



# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
- Operador de cruce OX1



# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
  - Operador de cruce OX2. Es una modificación del OX1 en el que se escogen al azar varios genes de uno de los padres para imponer en el otro parentel el orden de los elementos en las posiciones seleccionadas

Padre 1

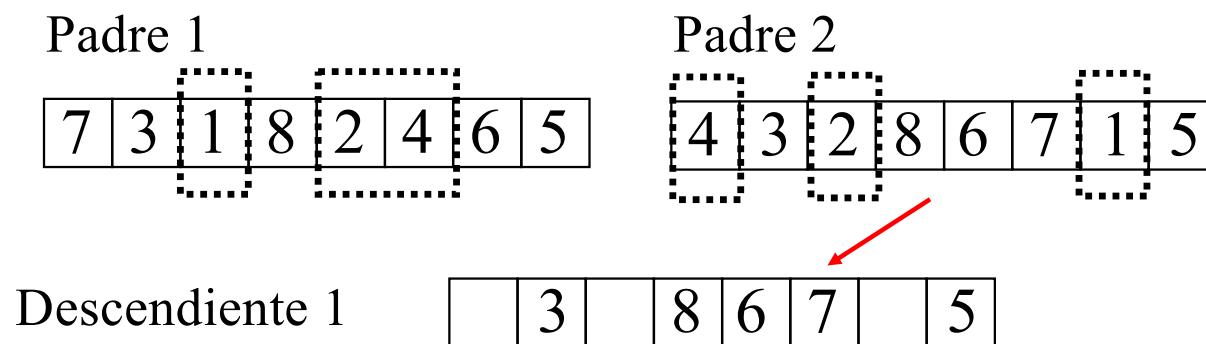
7	3	1	8	2	4	6	5
.....	.....	.....	.....	.....	.....	.....	.....

Padre 2

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

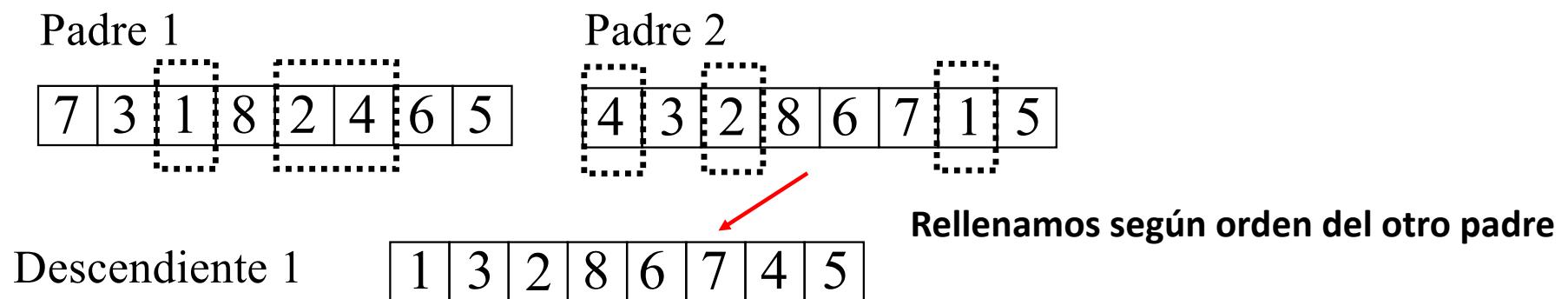
# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
    - Operador de cruce OX2. Es una modificación del OX1 en el que se escogen al azar varios genes de uno de los padres para imponer en el otro parentel el orden de los elementos en las posiciones seleccionadas



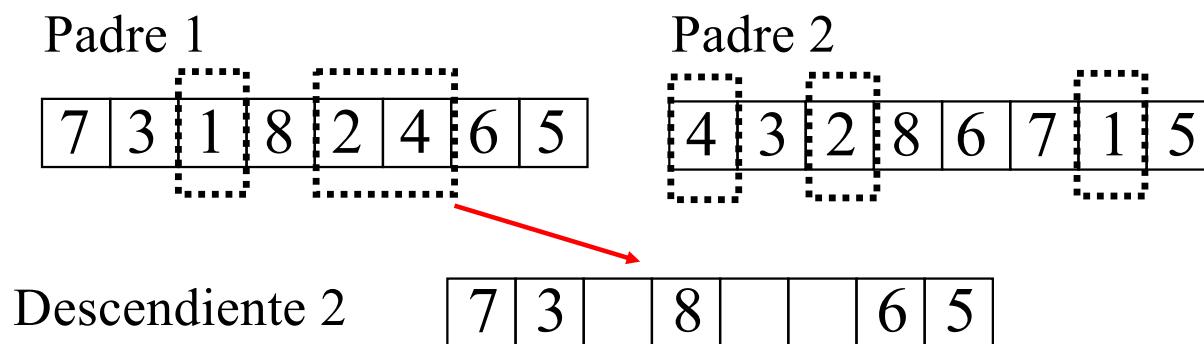
# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
  - Operador de cruce OX2. Es una modificación del OX1 en el que se escogen al azar varios genes de uno de los padres para imponer en el otro parentel el orden de los elementos en las posiciones seleccionadas



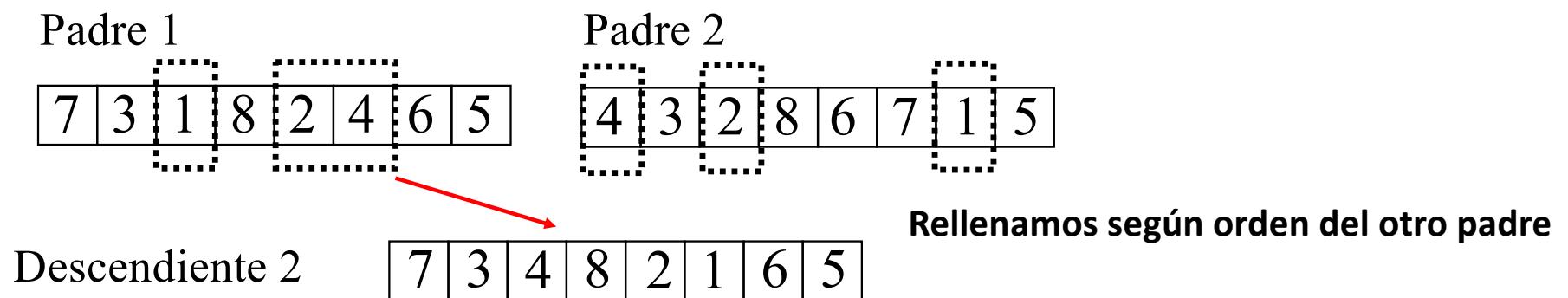
# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
  - Operador de cruce OX2. Es una modificación del OX1 en el que se escogen al azar varios genes de uno de los padres para imponer en el otro parentel el orden de los elementos en las posiciones seleccionadas



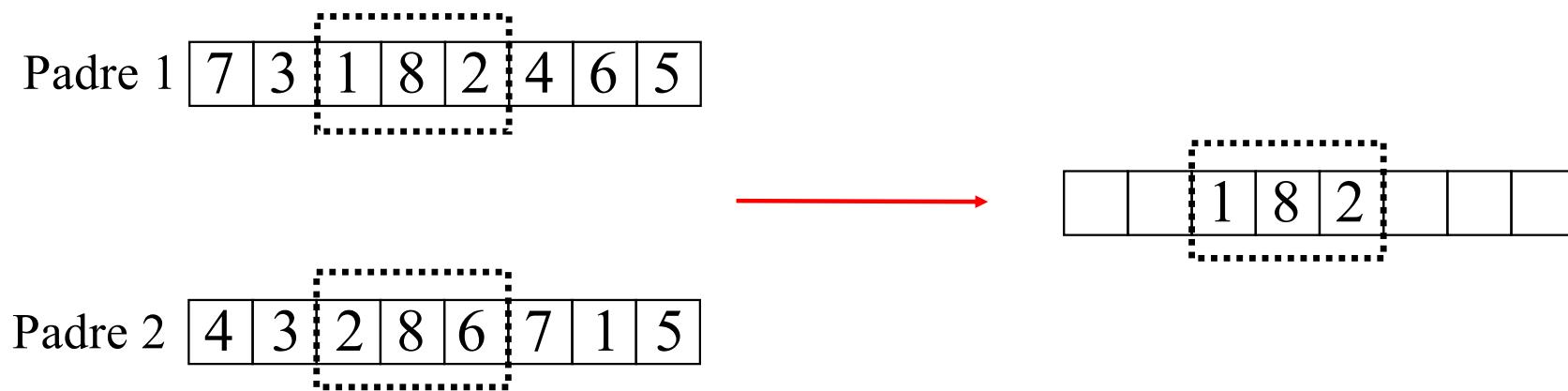
# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
  - Operador de cruce OX2. Es una modificación del OX1 en el que se escogen al azar varios genes de uno de los padres para imponer en el otro parentel el orden de los elementos en las posiciones seleccionadas



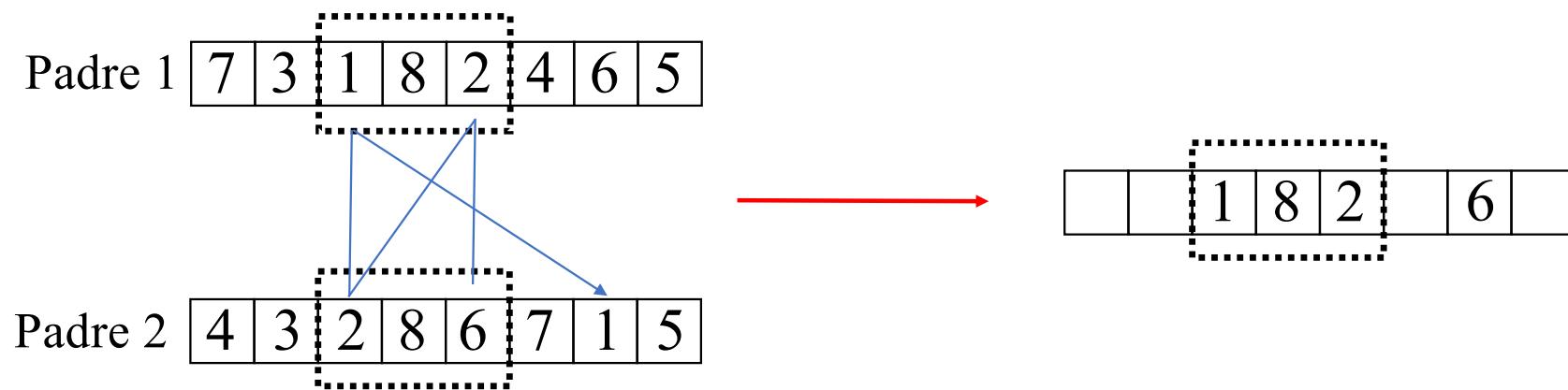
# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
  - Operador de cruce PMX (Partially Mapped Crossover)



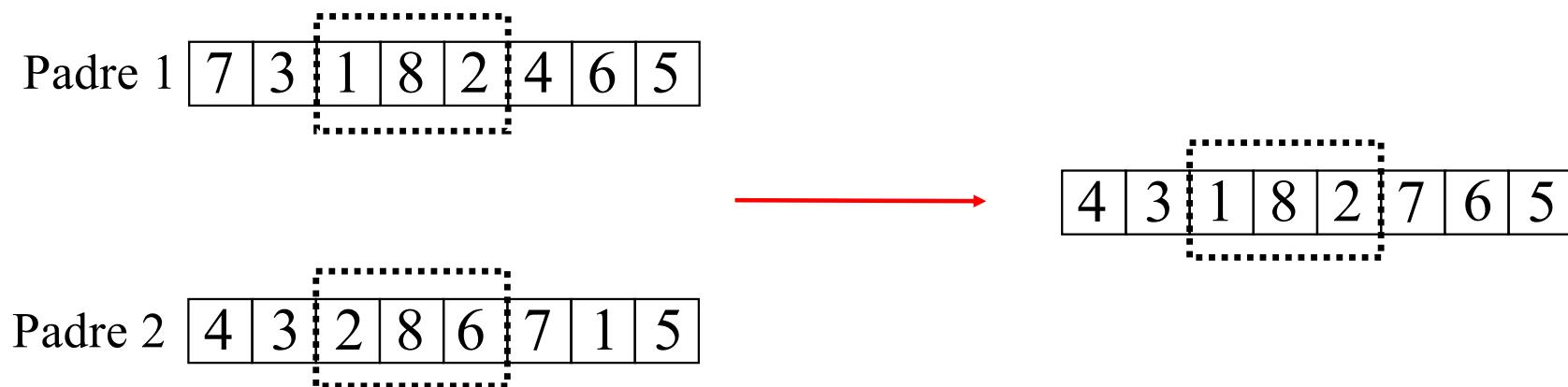
# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
  - Operador de cruce PMX (Partially Mapped Crossover)



# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
  - Operador de cruce PMX (Partially Mapped Crossover)

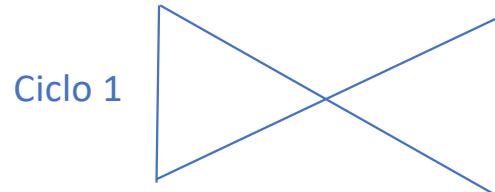


# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
  - Operador de cruce por ciclos

Padre 1 

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

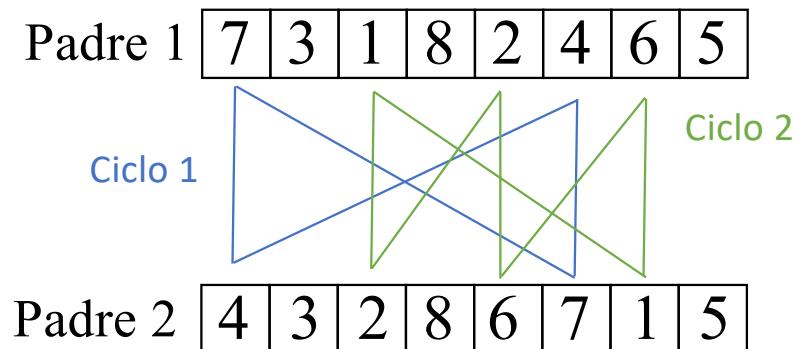


Padre 2 

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

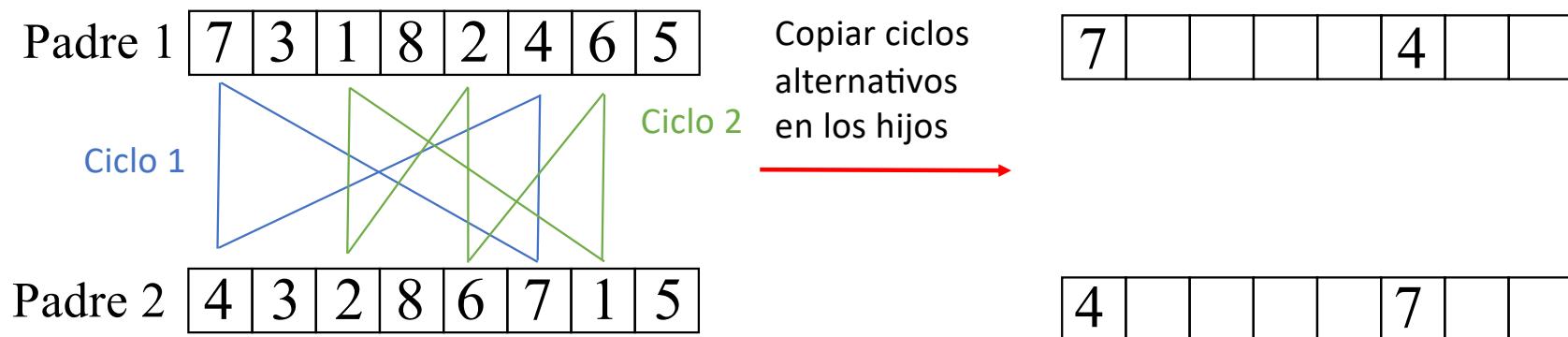
# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
  - Operador de cruce por ciclos



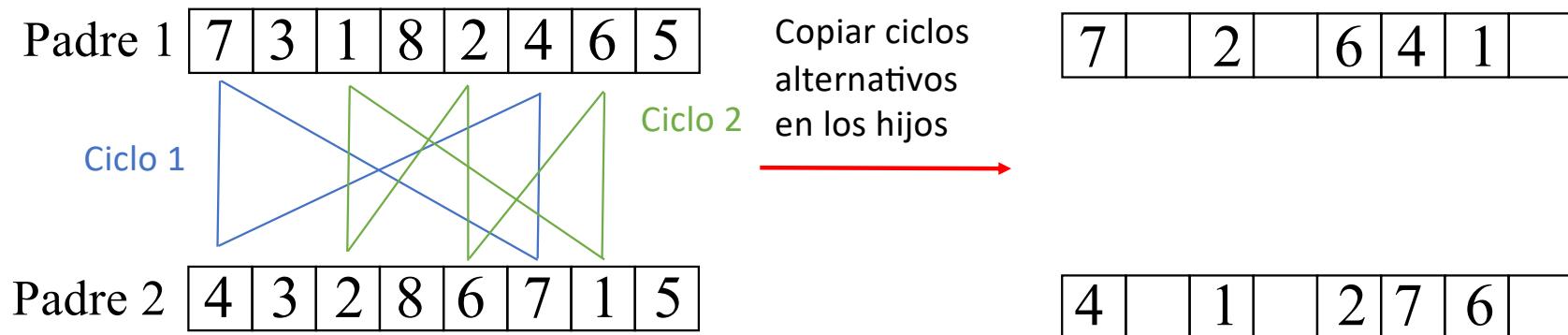
# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
  - Operador de cruce por ciclos



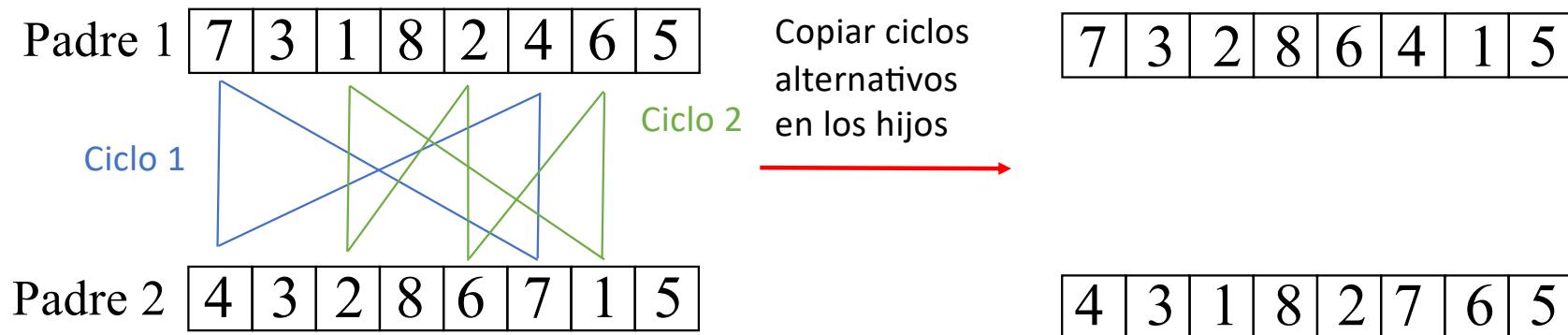
# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
  - Operador de cruce por ciclos



# Algoritmos genéticos: Operadores de cruce

- **Cruces para permutaciones** (representación ordinal)
  - Operador de cruce por ciclos



# Algoritmos genéticos

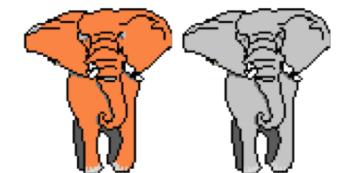


# Índice

- Introducción a la computación evolutiva
- **Algoritmos genéticos**
  - Modelos
  - Primeros pasos en la construcción de un AG
  - Estrategias de selección
  - Operadores de cruce
  - **Operadores de mutación**

# Algoritmos genéticos: Operadores de mutación

- Podría tenerse uno o más operadores de mutación para una representación
- Algunos aspectos importantes:
  - Debe permitir alcanzar cualquier parte del espacio de búsqueda
  - El tamaño de la mutación debe ser controlado
  - Se deben producir cromosomas válidos
  - Se aplica con una probabilidad muy baja
- **Operadores de mutación** más comunes:
  - Mutación clásica/estándar
  - Mutación uniforme
  - Mutación basada en intercambios y reordenación
  - Mutación basada en inserción
  - Mutación basada en inversión



# Algoritmos genéticos: Operadores de mutación

- **Mutación clásica/stándar:**

- Se selecciona uno o varios genes y se cambian sus valores
  - Un único gen



- Múltiples genes



- Cada gen se altera con una probabilidad  $p_m$



# Algoritmos genéticos: Operadores de mutación

- **Mutación uniforme:**

- Se selecciona uno gen y su valore (entero o real) es escogido aleatoriamente entre los valores de un rango



# Algoritmos genéticos: Operadores de mutación

- **Mutación basada en intercambios y reordenación:**

- Intercambios: se seleccionan dos genes y se intercambian sus valores

<table border="1"><tr><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td></tr></table>	1	0	1	1	0	1	→	<table border="1"><tr><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td></tr></table>	1	1	1	0	0	1
1	0	1	1	0	1									
1	1	1	0	0	1									

<table border="1"><tr><td>1</td><td>2</td><td>5</td><td>4</td><td>3</td><td>6</td></tr></table>	1	2	5	4	3	6	→	<table border="1"><tr><td>1</td><td>4</td><td>5</td><td>2</td><td>3</td><td>6</td></tr></table>	1	4	5	2	3	6
1	2	5	4	3	6									
1	4	5	2	3	6									

- Reordenación: se selecciona un subconjunto de genes y sus valores son reordenados de manera aleatoria

<table border="1"><tr><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td></tr></table>	1	0	1	1	0	1	→	<table border="1"><tr><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td></tr></table>	1	1	1	0	0	1
1	0	1	1	0	1									
1	1	1	0	0	1									

<table border="1"><tr><td>1</td><td>2</td><td>5</td><td>4</td><td>3</td><td>6</td></tr></table>	1	2	5	4	3	6	→	<table border="1"><tr><td>1</td><td>4</td><td>2</td><td>5</td><td>3</td><td>6</td></tr></table>	1	4	2	5	3	6
1	2	5	4	3	6									
1	4	2	5	3	6									

# Algoritmos genéticos: Operadores de mutación

- **Mutación basada en inserción:**

- Selecciona aleatoriamente un gen del genotipo y lo inserta en otro lugar al azar



- Puede realizarse para un conjunto de genes



# Algoritmos genéticos: Operadores de mutación

- **Mutación basada en inversión:**

- Selecciona un conjunto de genes del genotipo y los inserta de manera inversa (como la mutación por reordenación pero manteniendo cierto orden)



- Puede realizarse insertándola en otra posición



# Algoritmos genéticos



# Índice

- Introducción a la computación evolutiva
- **Algoritmos genéticos**
  - Modelos
  - Primeros pasos en la construcción de un AG
  - Estrategias de selección
  - Operadores de cruce
  - Operadores de mutación
  - **¿Cruce o mutación?**

# Algoritmos genéticos: ¿Cruce o mutación?

- Depende del problema que se desea resolver pero, en general, suele ser bueno **utilizar ambos operadores**
- Se puede utilizar un **algoritmo que sólo emplee mutación**, pero **no un algoritmo que sólo use cruce**
- **Cruce vs Mutación:**
  - **Cruce:** es un método para compartir información entre cromosomas que juega un papel central en los algoritmos genéticos ya que explota la información disponible de muestras previas para influir en futuras búsquedas
    - Sólo el cruce puede combinar lo mejor de dos padres
  - **Mutación:** es un método para explorar el espacio de búsqueda con el objetivo de descubrir áreas prometedoras lejos de los padres
    - Sólo la mutación puede introducir nuevos genes en la población

# Índice

- Introducción a la computación evolutiva
- **Algoritmos genéticos**
  - Modelos
  - Primeros pasos en la construcción de un AG
  - Estrategias de selección
  - Operadores de cruce
  - Operadores de mutación
  - ¿Cruce o mutación?
  - **Estrategias de reemplazo**

# Algoritmos genéticos: Estrategias de reemplazo

- La **presión selectiva se ve afectada** por la forma en que los cromosomas de la población son **reemplazados por los nuevos descendientes**
- Se pueden utilizar métodos de reemplazo aleatorios, o determinísticos
- **Elitismo: no reemplazar al mejor cromosoma** de la población. Su uso está aconsejado en modelos generacionales para no perder la mejor solución encontrada
- En los **modelos estacionarios** (no se reemplaza la población completa) nos encontramos con diferentes propuestas de reemplazo:
  - Reemplazar al peor de la población
  - Torneo restringido
  - Peor entre semejantes
  - *Crowding* determinístico
  - Usar técnicas de selección de padres como técnicas de reemplazo

# Algoritmos genéticos: Estrategias de reemplazo

- **Reemplazar al peor de la población:** genera alta presión selectiva
- **Torneo restringido:** se reemplaza al más parecido de entre N individuos. Mantiene una cierta diversidad
- **Peor entre semejantes:** se parte de un descendiente generado y éste reemplaza al peor individuo de un conjunto de los N más parecidos (padres o individuos de la población anterior). Busca un equilibrio entre diversidad y presión selectiva
- ***Crowding* determinístico:** el hijo reemplaza a su parente más parecido. Mantiene diversidad

# Algoritmos genéticos



# Índice

- Introducción a la computación evolutiva
- **Algoritmos genéticos**
  - Modelos
  - Primeros pasos en la construcción de un AG
  - Estrategias de selección
  - Operadores de cruce
  - Operadores de mutación
  - ¿Cruce o mutación?
  - Estrategias de reemplazo
  - **Consideraciones finales**

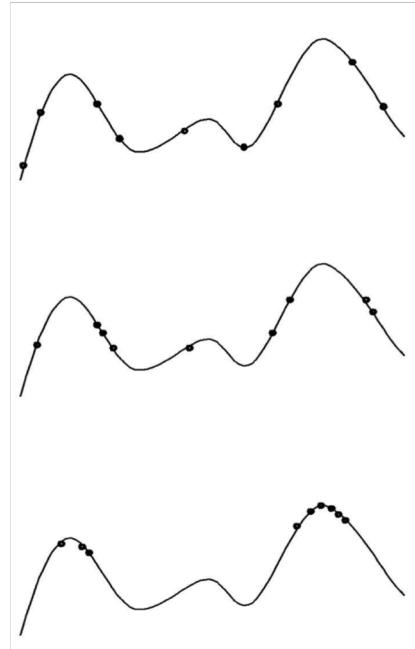
# Algoritmos genéticos: Consideraciones finales

- **Criterio de parada**

- Al alcanzar el óptimo
- Recursos limitados de CPU
  - Fijar el número máximo de generaciones
  - Fijar el número máximo de iteraciones
- Basado en la evolución
  - Después de algunas iteraciones sin mejora

# Algoritmos genéticos: Consideraciones finales

- Fases de un algoritmo genético *ideal*



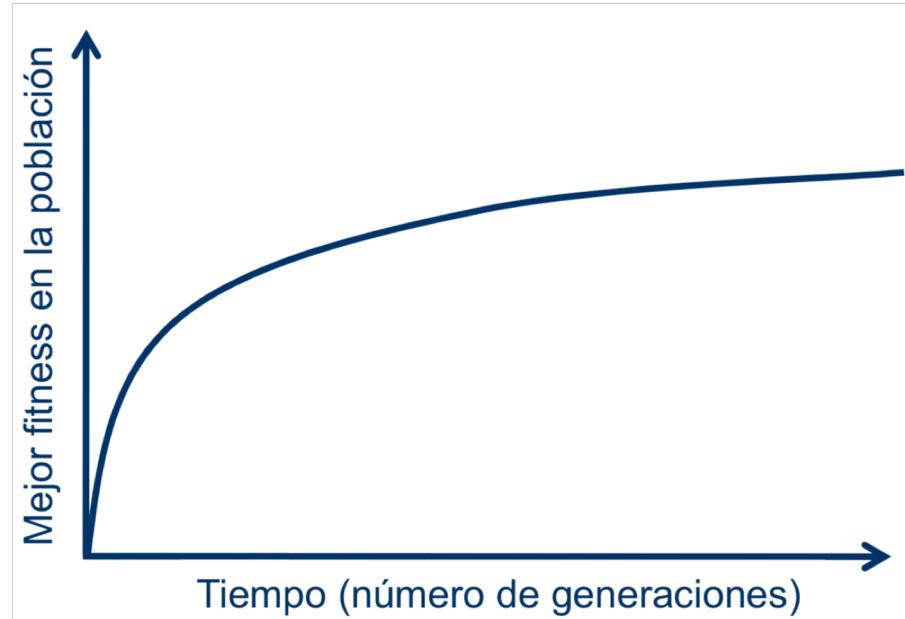
**Generaciones iniciales:** Distribución aleatoria de la población

**Generaciones intermedias:** Población cerca de las colinas

**Generaciones finales:** Población concentrada en las cimas de las colinas más altas

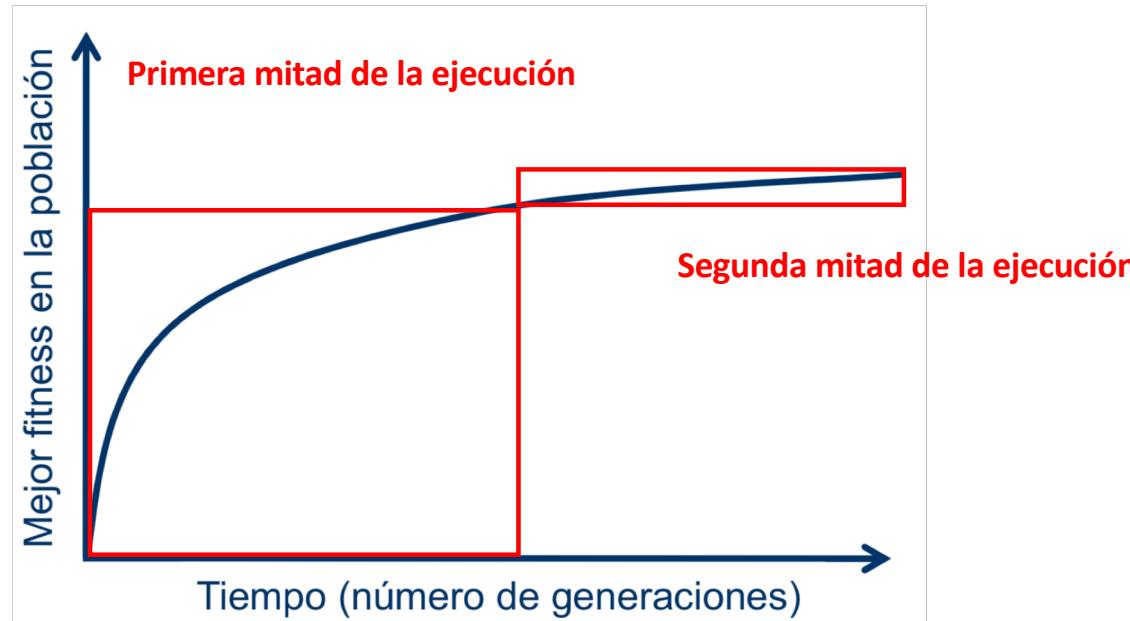
# Algoritmos genéticos: Consideraciones finales

- **Evolución típica del fitness** durante la ejecución de un algoritmo genético



# Algoritmos genéticos: Consideraciones finales

- ¿Merece la pena una ejecución larga? depende



# Algoritmos genéticos: Consideraciones finales

- **Nunca** sacar conclusiones de **una única ejecución**. Utilizar medidas estadísticas (medias, medianas, etc.) con un número suficiente de ejecuciones independientes
- No se debe comprobar la actuación de un algoritmo sobre ejemplos simples si se desea trabajar con casos reales
- Diferentes enfoques/diseños:
  - Encontrar **una solución muy buena** al menos una vez
  - Encontrar **al menos una solución muy buena** en cada ejecución

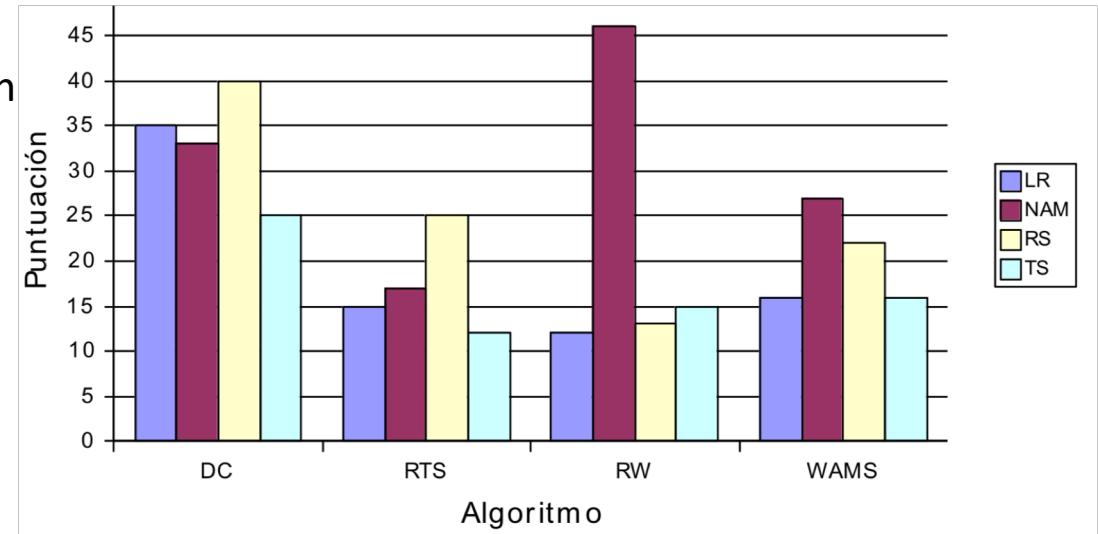
# Algoritmos genéticos: Consideraciones finales

- **Estudio comparativo de modelos estacionarios**
  - Diferentes combinaciones de selección-reemplazo para un algoritmo genético
  - Características comunes:
    - Cruce BLX-0.5
    - Mutación no uniforme con probabilidad 1/8
    - Tamaño de población: 60 individuos
    - 100,000 evaluaciones por ejecución
  - Evaluación sobre un conjunto de 13 problemas y medias de 50 ejecuciones
  - Se da un ranking en base a lo bueno/malo que es (menor valor significa peor resultado). Se suman los resultados para los 13 problemas

# Algoritmos genéticos: Consideraciones finales

- **Estudio comparativo de modelos estacionarios**

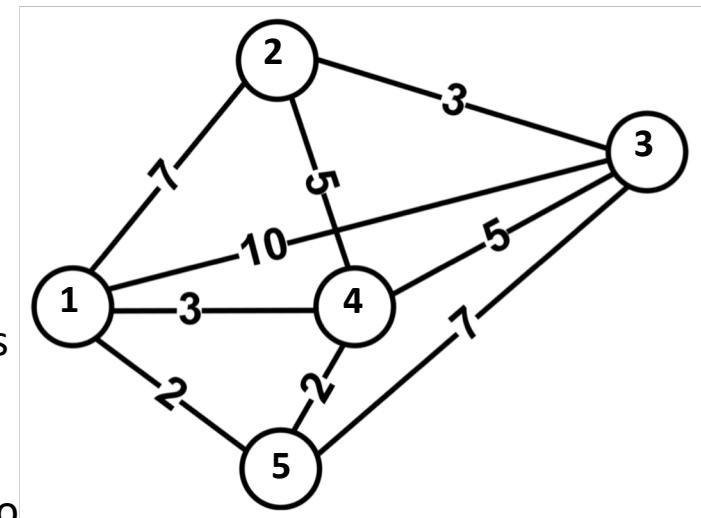
- LR: selección por ranking
- NAM: emparejamiento variado inversión
- RS: selección aleatoria
- TS: selección por torneos
- DC: *Crowding* determinístico
- RTS: Torneo restringido
- RW: Reemplazar al peor
- WAMS: Peor entre semejantes



# Algoritmos genéticos: Consideraciones finales

- **Ejercicio Clase. TSP (minimizar función objetivo)**

- Partimos de una representación ordinal con cuatro individuos:
  - (1 3 5 4 2) F=31
  - (3 1 2 4 5) F=31
  - (1 2 4 3 5) F=26
  - (5 4 2 1 3) F=31
- Aplicar un modelo generacional con elitismo (n=1)
  - Diferentes operadores de selección/cruce/mutación
- Aplicar un modelo estacionario con diferentes técnicas de reemplazo
  - Diferentes operadores de selección/cruce/mutación
- Criterio de parada: 5 generaciones ó alcanzar el óptimo



# Índice

- Introducción a la computación evolutiva
- Algoritmos genéticos
- **Diversidad Vs Convergencia**

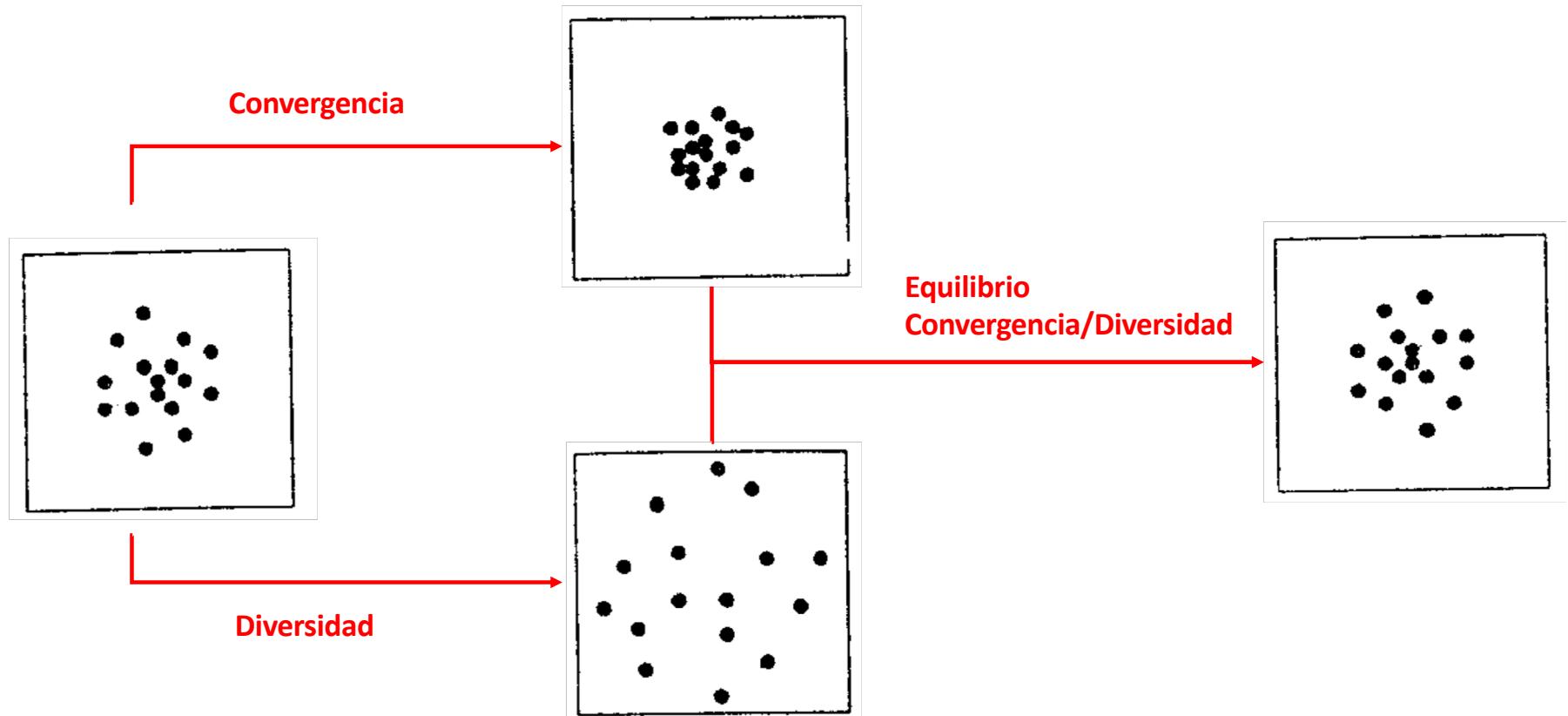
# Algoritmos genéticos: Diversidad Vs Convergencia

- Todo algoritmo de búsqueda necesita establecer un **equilibrio entre dos factores aparentemente contrapuestos:**
  - Exploración del espacio de soluciones, para realizar una búsqueda en amplitud, localizando así zonas prometedoras
  - Explotación espacio de búsqueda, para hacer una búsqueda en profundidad en dichas zonas, obteniendo así las mejores soluciones
- Los algoritmos genéticos son un tipo de algoritmo de búsqueda de propósito general, cuyos operadores pueden establecer un equilibrio adecuado entre exploración y explotación

# Algoritmos genéticos: Diversidad Vs Convergencia

- Dos factores contrapuestos influyen sobre la efectividad de un algoritmo genético:
  - **Convergencia:** Centrar la búsqueda en regiones prometedoras mediante presión selectiva
    - **Presión selectiva:** permite que los mejores individuos sean seleccionados para reproducirse.  
Es necesaria para que el proceso de búsqueda no sea aleatorio
  - **Diversidad:** Evitar la convergencia prematura (rápida convergencia hacia zonas que no contienen el óptimo global)

# Algoritmos genéticos: Diversidad Vs Convergencia



# Algoritmos genéticos: Diversidad Vs Convergencia

- La **diversidad** está asociada a las **diferencias entre los individuos** de la población
- **Falta de diversidad genética** significa que todos los **individuos en la población son parecidos**
  - Falta de diversidad implica una convergencia prematura hacia óptimos locales
- **Soluciones:**
  - Inclusión de mecanismos de diversidad en la evolución
  - Reinicialización cuando se produce convergencia prematura

# Algoritmos genéticos: Diversidad Vs Convergencia

- Inclusión de diversidad en la evolución:
  - Diversidad con la mutación
  - Diversidad con el cruce
  - Separación espacial
  - Adaptación, auto-adaptación, metaevolución
  - Estrategias de reemplazamiento

# Algoritmos genéticos: Diversidad Vs Convergencia

- Inclusión de diversidad en la evolución:
  - **Diversidad con la mutación**
  - Diversidad con el cruce
  - Separación espacial
  - Adaptación, auto-adaptación, metaevolución
  - Estrategias de reemplazamiento

# Algoritmos genéticos: Diversidad Vs Convergencia

- **Diversidad con la mutación**

- Una probabilidad de mutación alta no soluciona la convergencia prematura
- La solución consiste en adaptar dicha probabilidad de mutación
  - Reducirla durante la ejecución: primero diversidad, después convergencia
  - Aplicar probabilidades altas sobre soluciones malas y pequeña sobre las buenas

# Algoritmos genéticos: Diversidad Vs Convergencia

- Inclusión de diversidad en la evolución:
  - Diversidad con la mutación
  - **Diversidad con el cruce**
  - Separación espacial
  - Adaptación, auto-adaptación, metaevolución
  - Estrategias de reemplazamiento

# Algoritmos genéticos: Diversidad Vs Convergencia

- **Diversidad con el cruce**

- Generar diversidad mediante el cruce depende de los siguientes factores:
  - Técnicas de emparejamiento
  - Técnicas para generar los hijos
  - Número de padres
  - Número de hijos

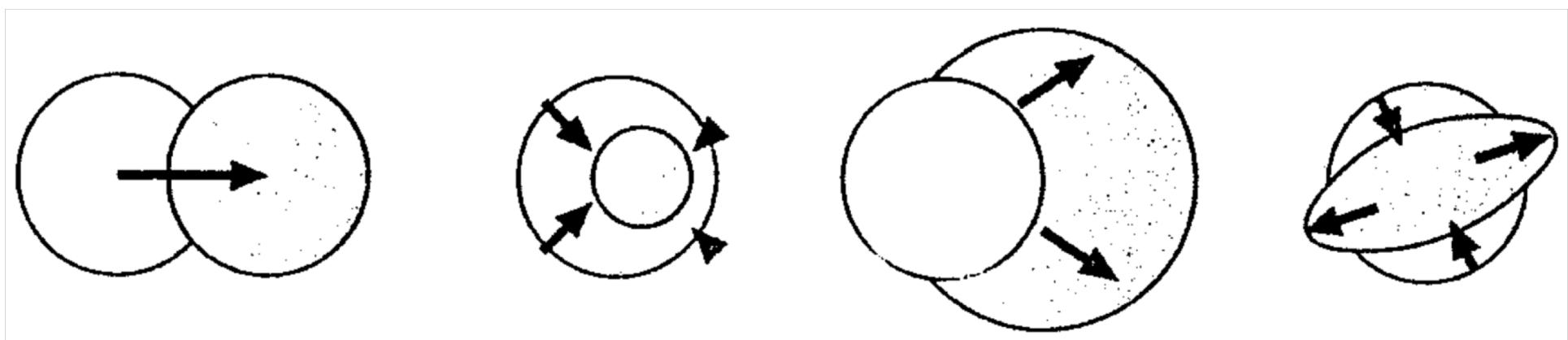
# Algoritmos genéticos: Diversidad Vs Convergencia

- **Diversidad con el cruce: Técnicas de emparejamiento.** Los padres se pueden seleccionar de forma que se mantenga la diversidad de la población
  - **Prohibición de cruce basada en ascendencia.** Un individuo no puede emparejarse con él mismo, ni con sus padres, ni con sus hijos, ni con sus hermanos
  - **Prohibición de incesto.** Dos padres se cruzan si su distancia *Hamming* está por encima de cierto umbral
  - **Emparejamiento variado.** Un cromosoma se cruza con otro que es bastante diferente

# Algoritmos genéticos: Diversidad Vs Convergencia

- Diversidad con el cruce: Técnicas para generar los hijos.

Equilibrio entre las siguientes características



Traslación

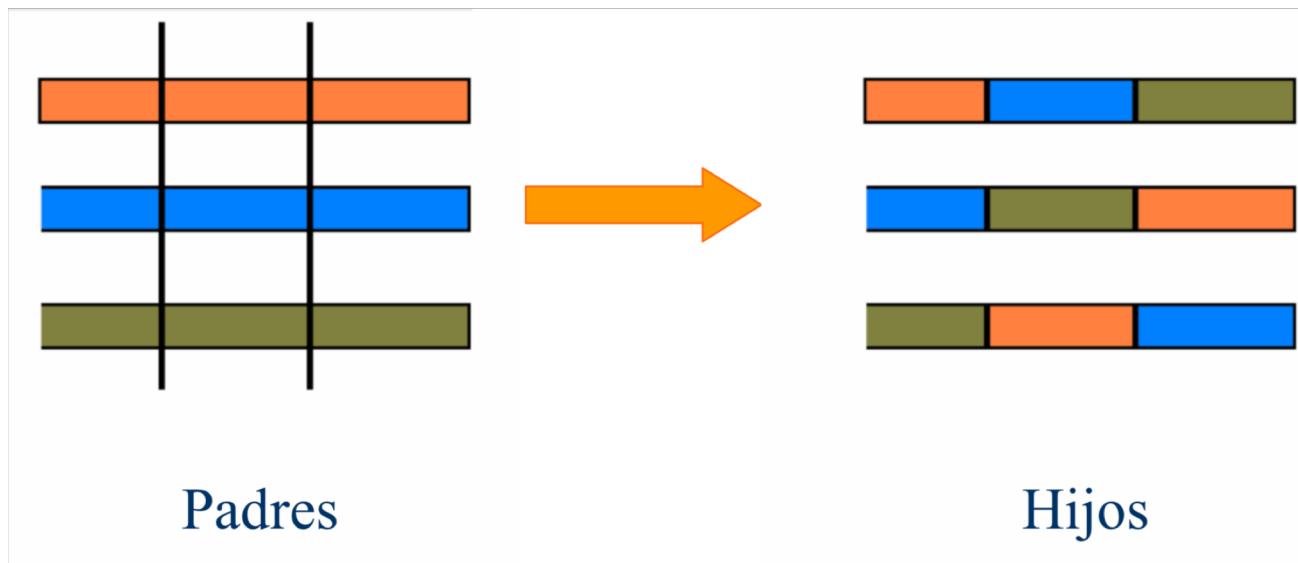
Concentración

Ampliación

Distribución de dirección

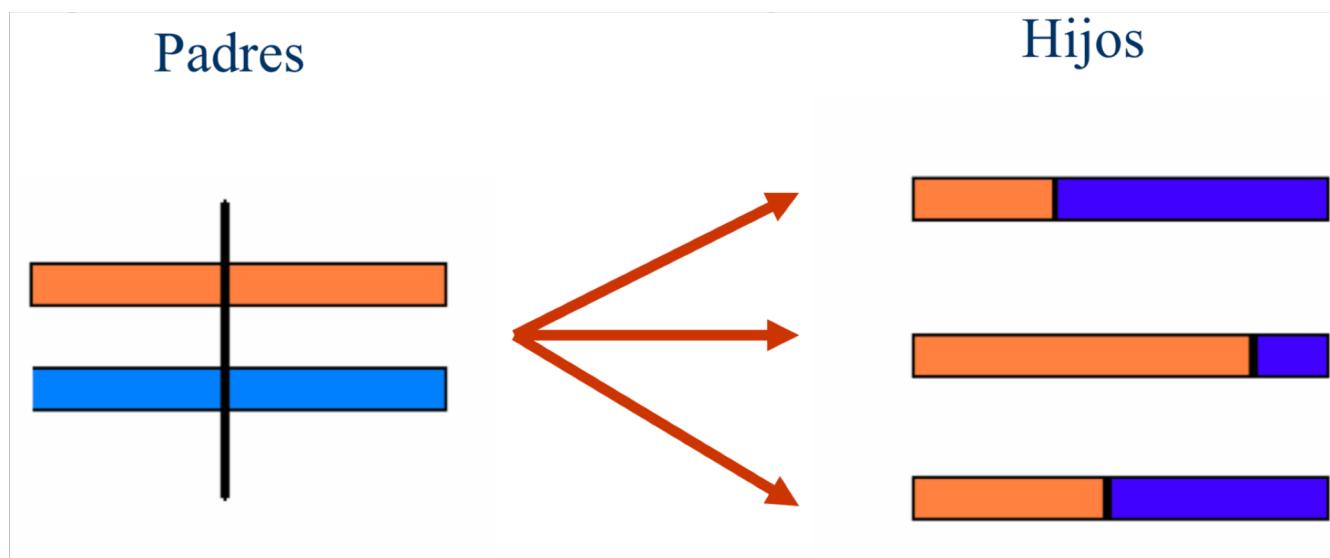
# Algoritmos genéticos: Diversidad Vs Convergencia

- Diversidad con el cruce: Número de padres
  - Operadores de cruce multi-padres



# Algoritmos genéticos: Diversidad Vs Convergencia

- Diversidad con el cruce: Número de hijos
  - Operadores de cruce multi-hijos



# Algoritmos genéticos: Diversidad Vs Convergencia

- Inclusión de diversidad en la evolución:
  - Diversidad con la mutación
  - Diversidad con el cruce
  - **Separación espacial**
  - Adaptación, auto-adaptación, metaevolución
  - Estrategias de reemplazamiento

# Algoritmos genéticos: Diversidad Vs Convergencia

- **Separación espacial**
- Los métodos de preservación de la diversidad basados en separación espacial han sido propuestos para evitar la convergencia prematura. Los más representativos son los **algoritmos genéticos distribuidos** y los **algoritmos genéticos celulares**
  - **Algoritmos genéticos distribuidos:** la población se divide en subpoblaciones que son tratadas como islas independientes en cada una de las cuales se ejecuta una instancia distinta del algoritmo. Estas instancias se comunican entre sí mediante unas cadenas de conexión
  - **Algoritmos genéticos celulares:** una única instancia del algoritmo genético opera sobre la población en forma de vecindarios; es decir, para cada individuo, realiza las operaciones correspondientes entre él y sus vecinos

# Algoritmos genéticos: Diversidad Vs Convergencia

- Inclusión de diversidad en la evolución:
  - Diversidad con la mutación
  - Diversidad con el cruce
  - Separación espacial
  - **Adaptación, auto-adaptación, metaevolución**
  - Estrategias de reemplazamiento

# Algoritmos genéticos: Diversidad Vs Convergencia

- **Adaptación, auto-adaptación, meta-evolución**
- **Adaptación:** ajuste de determinados elementos (parámetros o componentes) de los algoritmos genéticos a lo largo de la ejecución en función de su estado o información disponible sobre el espacio de búsqueda. Elementos:
  - Función de evaluación
  - Operadores genéticos
  - Representación
  - Operadores
  - Parámetros de control

# Algoritmos genéticos: Diversidad Vs Convergencia

- **Auto-adaptación:** evolución de los parámetros de acuerdo al comportamiento del algoritmos. Los parámetros suelen formar parte del cromosoma.
- **Meta-evolución:** considera la búsqueda del mejor AG para resolver un problema como un problema de optimización y se utiliza otro AG para resolverlo.

# Algoritmos genéticos: Diversidad Vs Convergencia

- Inclusión de diversidad en la evolución:
  - Diversidad con la mutación
  - Diversidad con el cruce
  - Separación espacial
  - Adaptación, auto-adaptación, metaevolución
  - **Estrategias de reemplazamiento**

# Algoritmos genéticos: Diversidad Vs Convergencia

- **Estrategias de reemplazamiento.** Introducir mayor diversidad o convergencia en la población en función del mecanismo de reemplazamiento de los cromosomas de la población en curso por los descendientes
- Son particularmente de interés en los modelos estacionarios, o cuando se introducen mecanismos de competición entre padres o hijos