

# Metaheurísticas

Unidad 3

Metaheurísticas basadas en Poblaciones

**Tema 2: Algoritmos Evolutivos**

# Objetivos

- Conocer el funcionamiento y estructura de los algoritmos evolutivos
- Conocer mecanismos básicos para adaptar la capacidad de exploración y explotación de los algoritmos evolutivos al problema a resolver
- Adquirir capacidad de diseñar y desarrollar un algoritmo evolutivo para resolver un problema dado

# Bibliografía

- [Fog98] D.B. Fogel (Ed.). *Evolutionary Computation. The Fossil Record. (Selected Readings on the History of Evolutionary Computation)*. IEEE Press, 1998.
- [ES03] A.E. Eiben, J.E. Smith. *Introduction to Evolutionary Computation*. Springer Verlag 2003.

# Índice

1. Introducción
2. Modelos de algoritmos evolutivos
3. Construcción y funcionamiento de un algoritmo evolutivo básico
4. Ejemplos de diseño de algoritmos evolutivo
5. Consideraciones finales

# Introducción

¿qué es un algoritmo evolutivo?

algoritmo de optimización, búsqueda y aprendizaje  
inspirado en los procesos de evolución natural y  
evolución genética

# Introducción

la evolución llevada a la computación



# Introducción

la evolución llevada a la computación

Los individuos de una especie existen y conviven dentro de una población

# Introducción

la evolución llevada a la computación

Los individuos de una especie existen y conviven dentro de una población

## REPRODUCCIÓN

Las generaciones contienen una generación padre, seguida por una generación hijos, que se considera descendiente

Cruce y mutación

# Introducción

la evolución llevada a la computación

Los individuos de una especie existen y conviven dentro de una población

## REPRODUCCIÓN

Las generaciones contienen una generación padre, seguida por una generación hijos, que se considera descendiente

Cruce y mutación

## SELECCIÓN

La adaptación (*fitness*) de cada individuo determina su probabilidad de supervivencia durante la selección

# Introducción

la evolución llevada a la computación

Los individuos de una especie existen y conviven dentro de una población

Los individuos que sobreviven se convierten en los padres de la próxima generación

## SELECCIÓN

La adaptación (*fitness*) de cada individuo determina su probabilidad de supervivencia durante la selección

## REPRODUCCIÓN

Las generaciones contienen una generación padre, seguida por una generación hijos, que se considera descendiente

Cruce y mutación

# Introducción

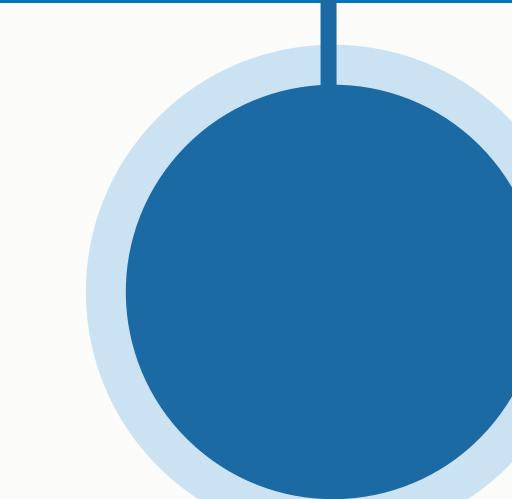
modelo evolutivo



# Introducción

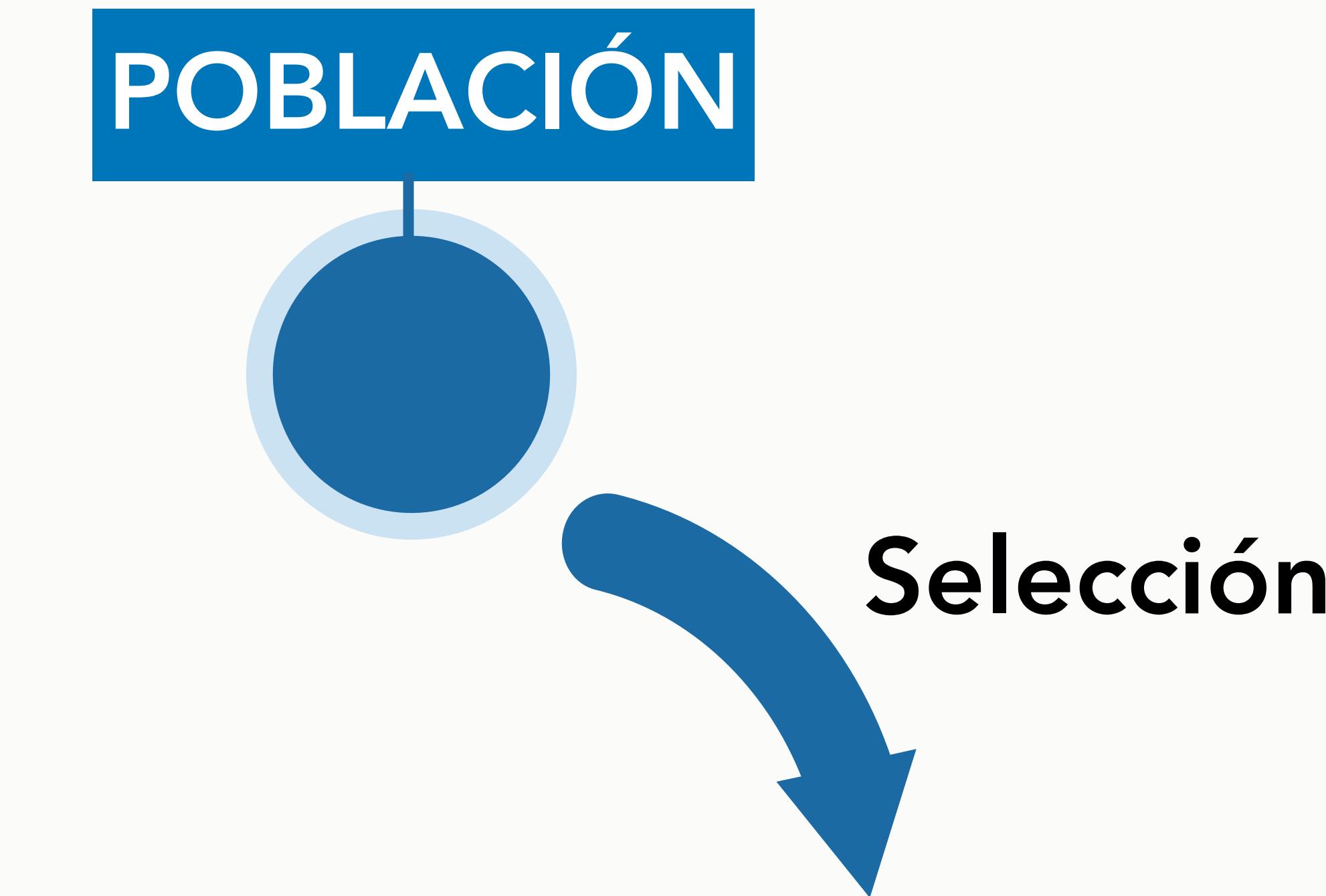
modelo evolutivo

**POBLACIÓN**



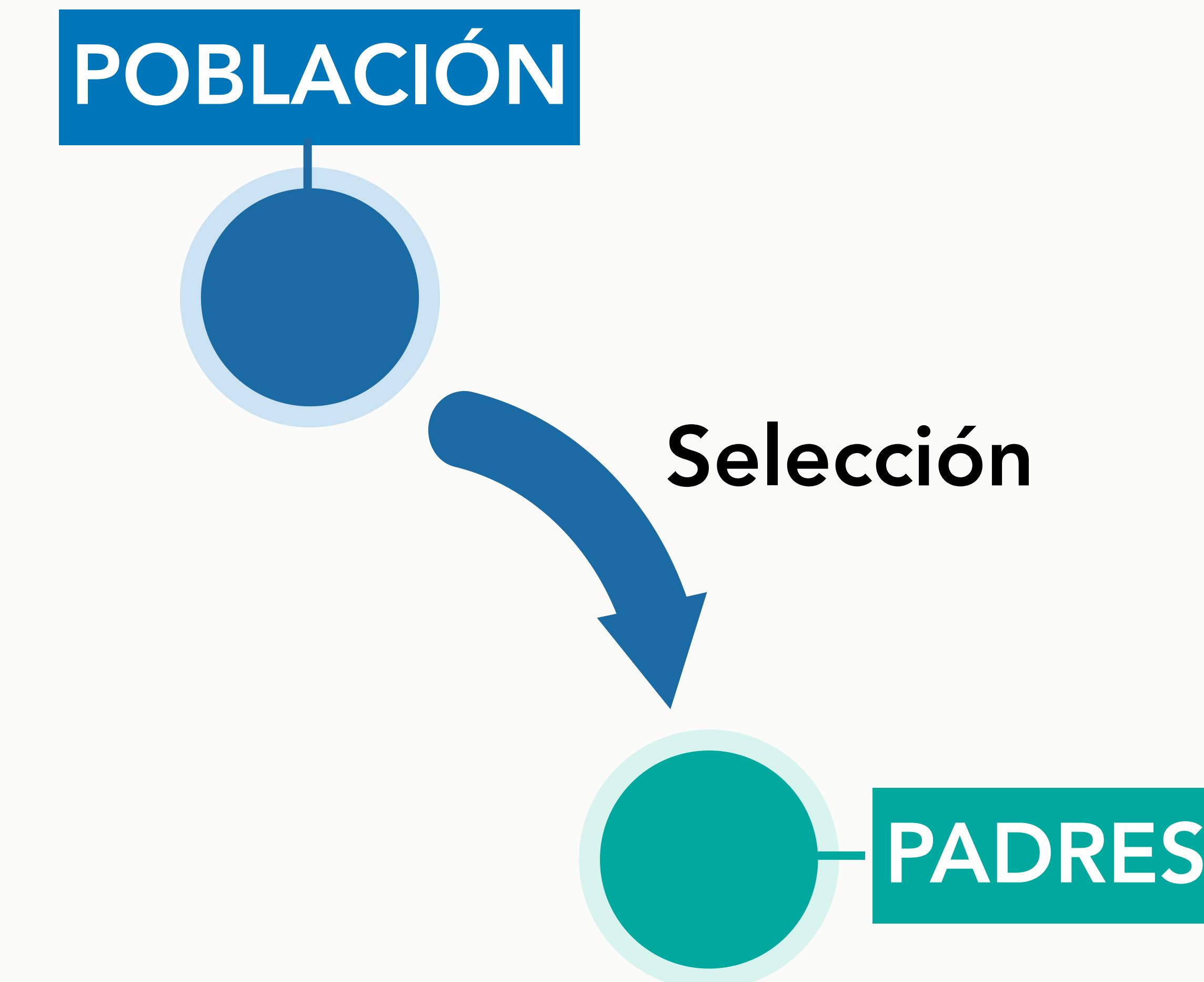
# Introducción

modelo evolutivo



# Introducción

modelo evolutivo



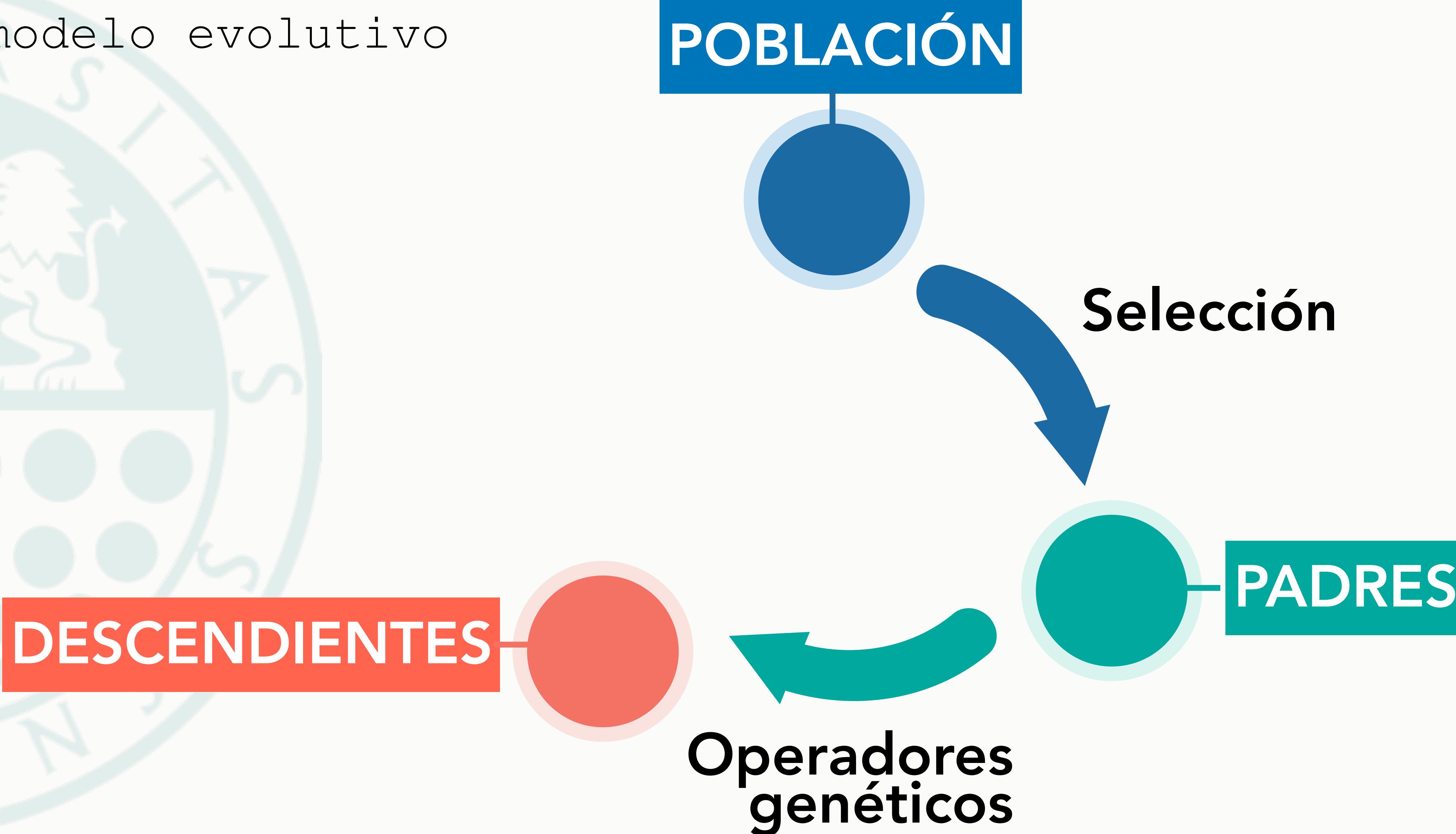
# Introducción

modelo evolutivo



# Introducción

modelo evolutivo



# Introducción

modelo evolutivo



# Introducción

estructura del algoritmo



## Introducción

estructura del algoritmo

```
INICIO
    t = 0
    inicializar P(t)
    evaluar P(t)
    MIENTRAS (no se cumpla la condición de parada) HACER
        t = t+1
        seleccionar P' desde P(t-1)
        recombinar P'
        mutar P'
        evaluar P(t)
        reemplazar P(t) a partir de P(t-1) y P'
    FIN MIENTRAS
FIN
```

## Introducción

metáfora

### Evolución natural

Evolución

Entorno

Individuo

*Fitness*

Gen

Alelo

### Optimización

Resolución de problemas

Problema de optimización

Solución candidata

Función objetivo

Elemento de la solución

Valor de un gen

Equivalencia entre el proceso evolutivo y el proceso de resolución de un problema de optimización

# Modelos evolutivos



# Modelos evolutivos

modelo generacional

En cada generación se crea una población con individuos nuevos

La nueva población reemplaza a la antigua

En algunos casos los mejores individuos de la anterior generación se incluyen en la de la siguiente >> **elitismo**

# Modelos evolutivos

modelo generacional

En cada generación se crea una población con individuos nuevos

La nueva población reemplaza a la antigua

En algunos casos los mejores individuos de la anterior generación se incluyen en la de la siguiente >> **elitismo**

modelo estacionario

En cada generación se seleccionan algunos padres de la población y se le aplican operadores genéticos

Estos descendientes reemplazan a individuos de la población

# Modelos evolutivos



- **Es un modelo elitista por definición**
- **Produce una presión selectiva alta, es decir una convergencia rápida cuando se reemplazan los peores**

# Modelos evolutivos

modelo generacional

En cada generación se crea una población con individuos nuevos

La nueva población reemplaza a la antigua

En algunos casos los mejores individuos de la anterior generación se incluyen en la de la siguiente >> **elitismo**

- **Es un modelo elitista por definición**
- **Produce una presión selectiva alta, es decir una convergencia rápida cuando se reemplazan los peores**

# Modelos evolutivos

modelo generacional

En cada generación se crea una población con individuos nuevos

La nueva población reemplaza a la antigua

En algunos casos los mejores individuos de la anterior generación se incluyen en la de la siguiente >> **elitismo**

modelo estacionario

- **Es un modelo elitista por definición**
- **Produce una presión selectiva alta, es decir una convergencia rápida cuando se reemplazan los peores**

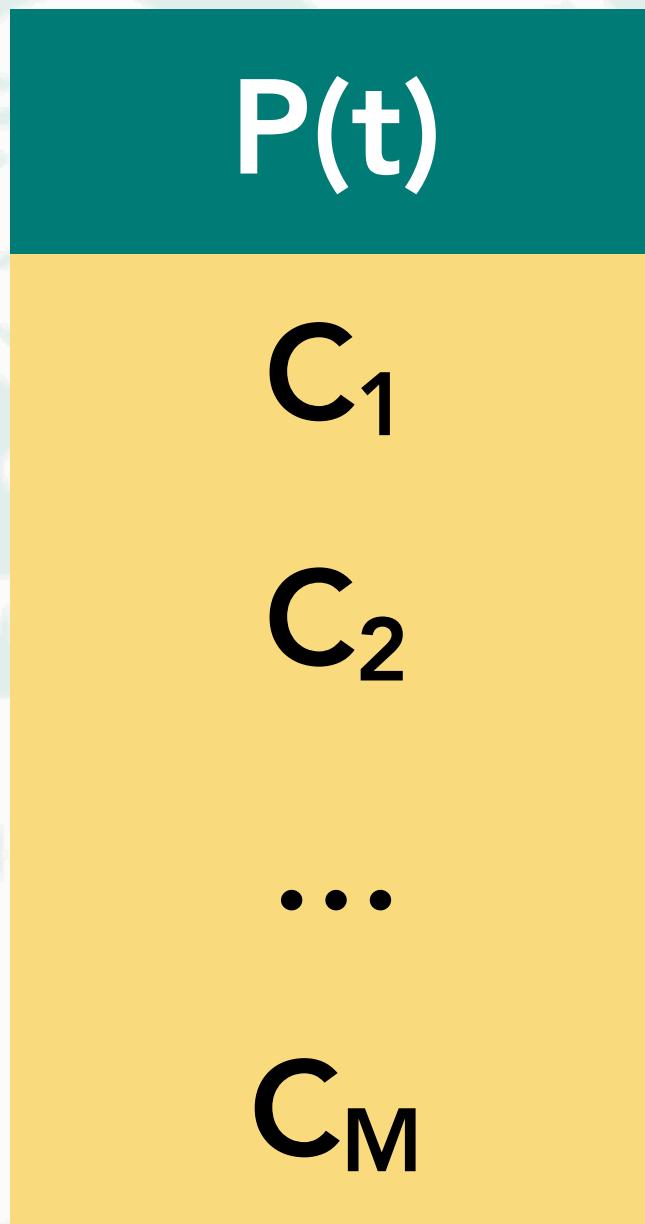
# Modelos evolutivos

modelo generacional



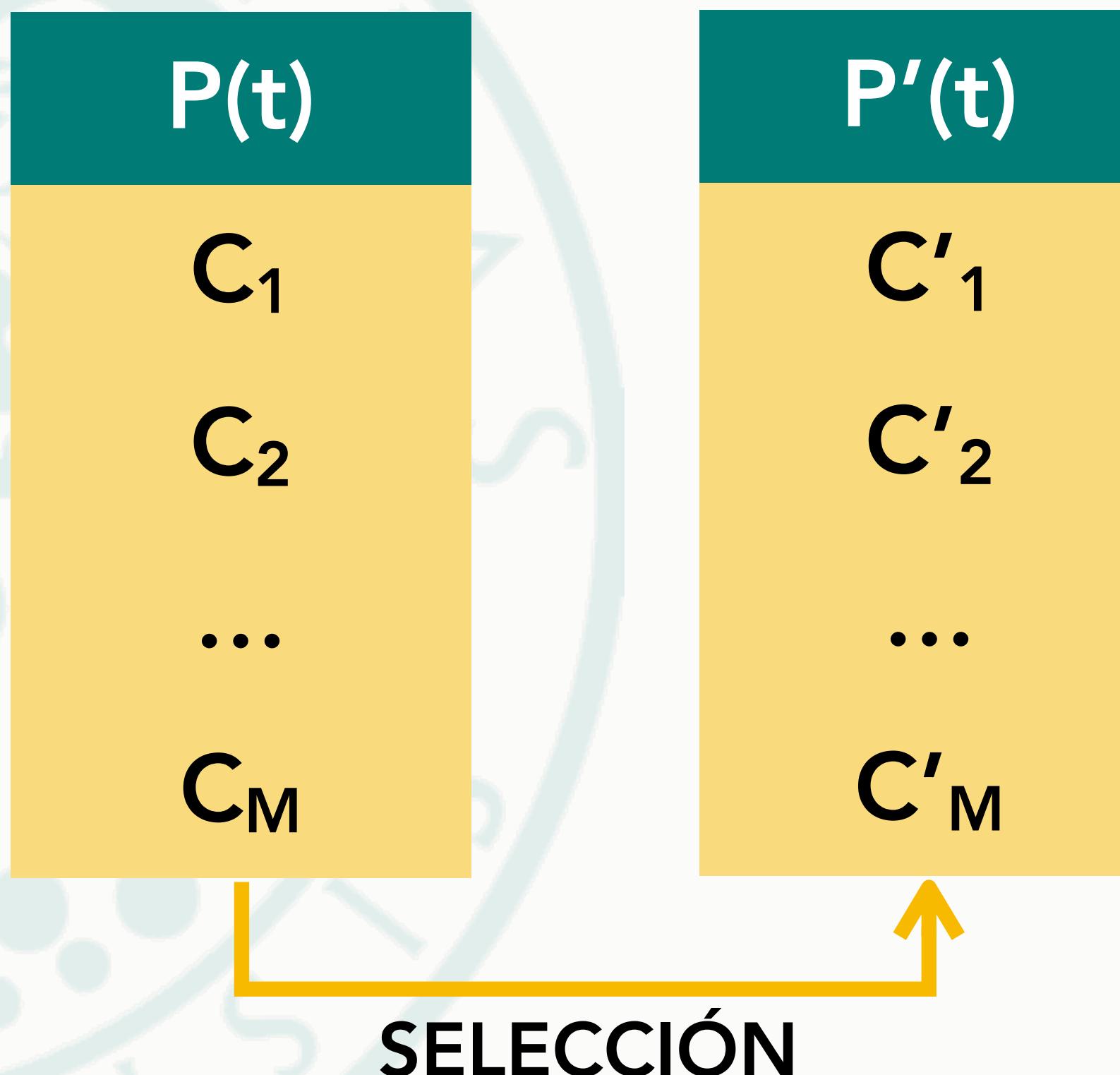
## Modelos evolutivos

modelo generacional



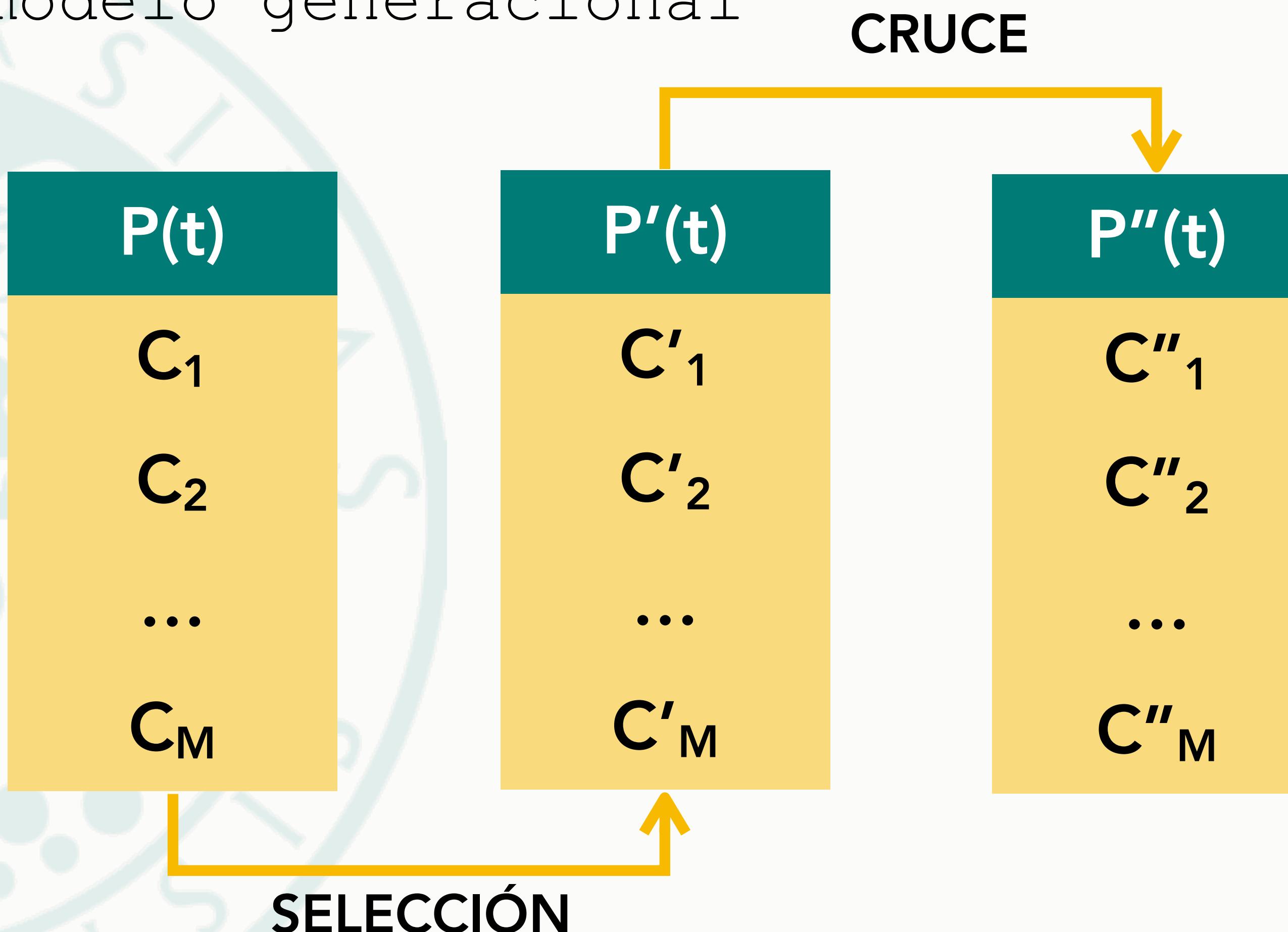
## Modelos evolutivos

modelo generacional



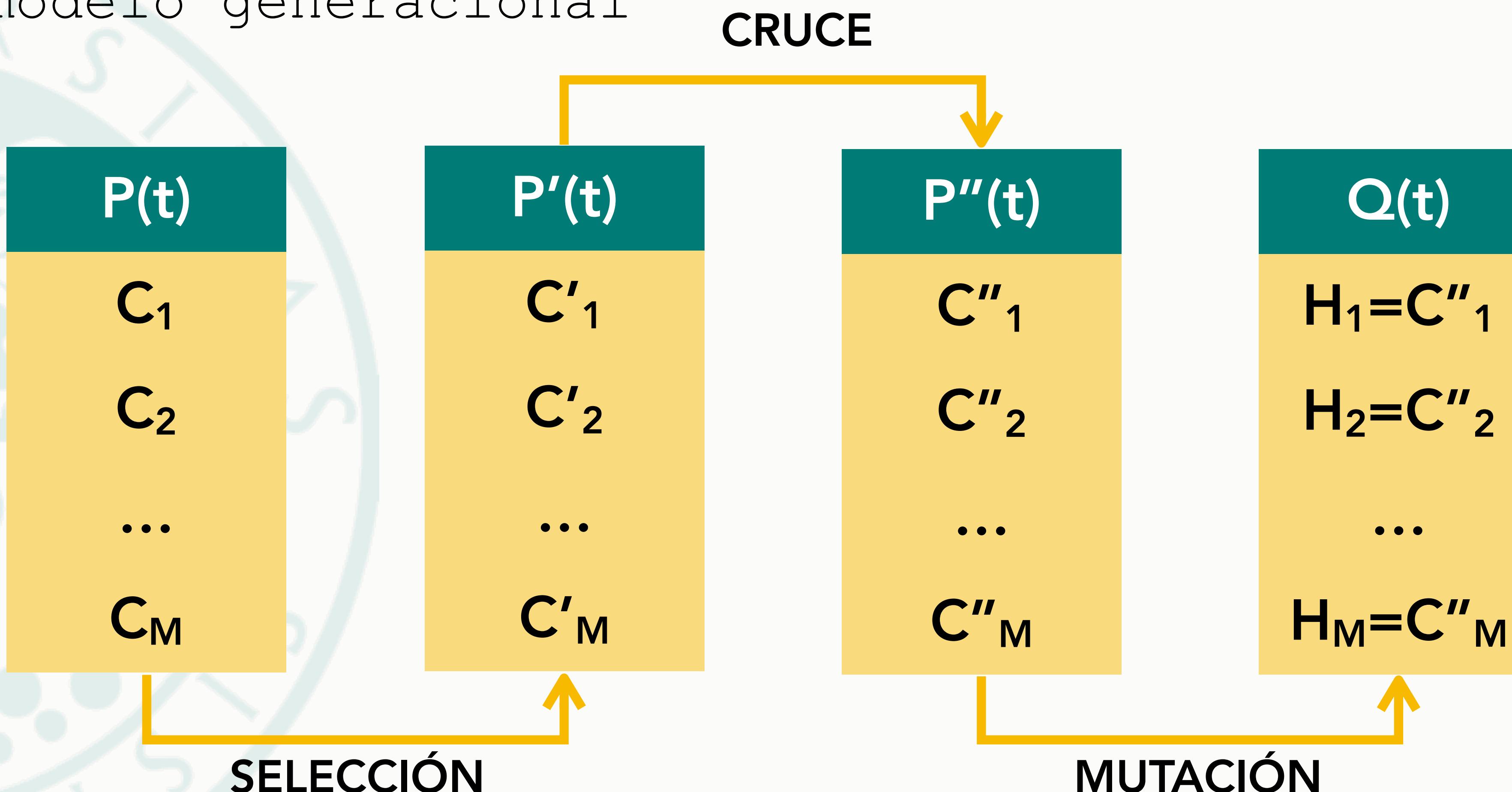
# Modelos evolutivos

modelo generacional



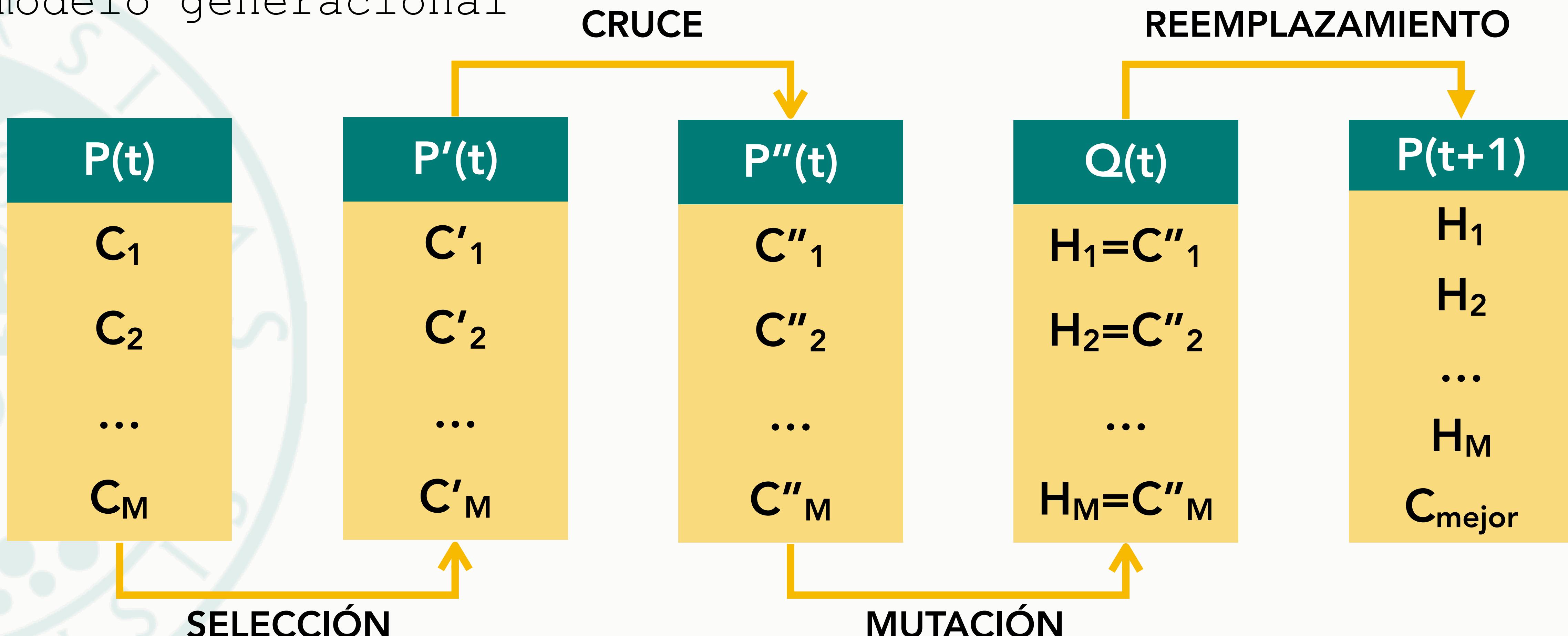
# Modelos evolutivos

modelo generacional



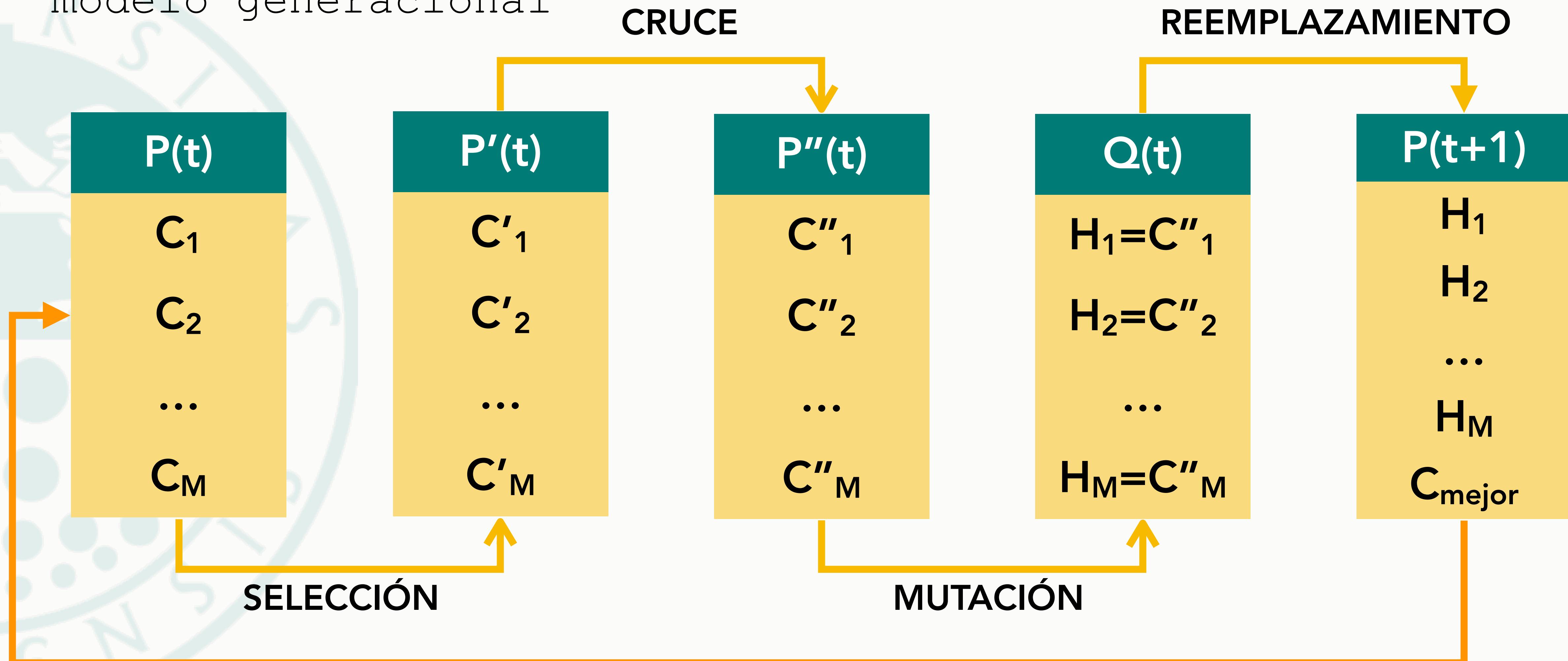
# Modelos evolutivos

modelo generacional



# Modelos evolutivos

modelo generacional



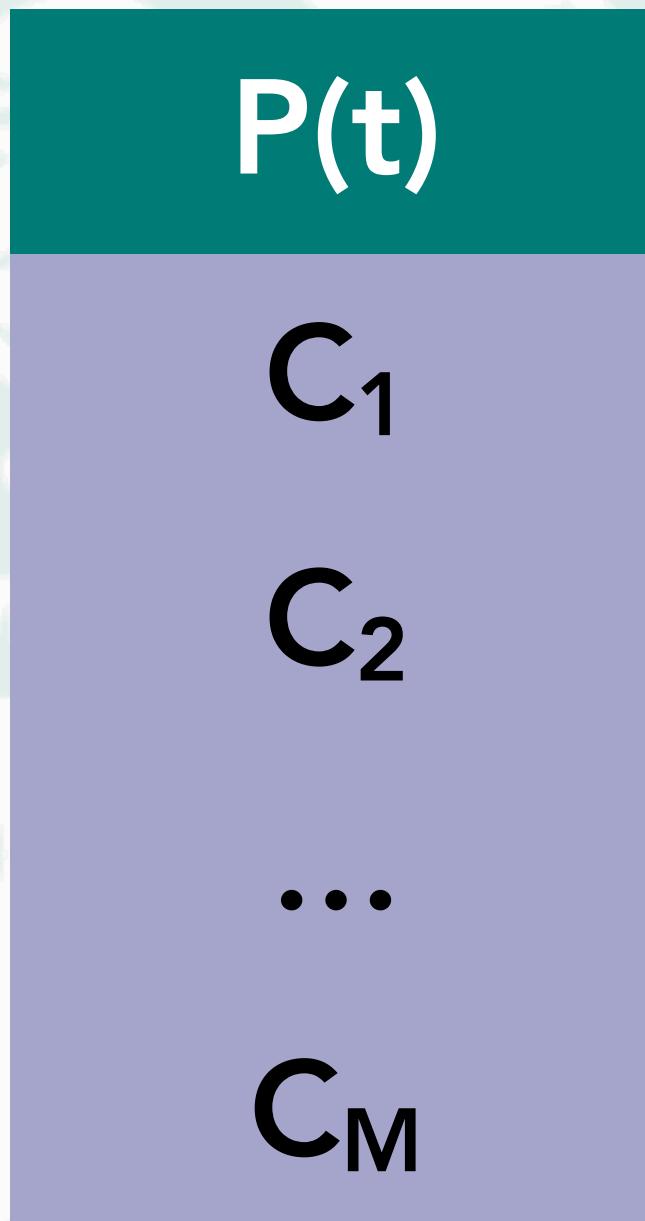
# Modelos evolutivos

modelo estacionario



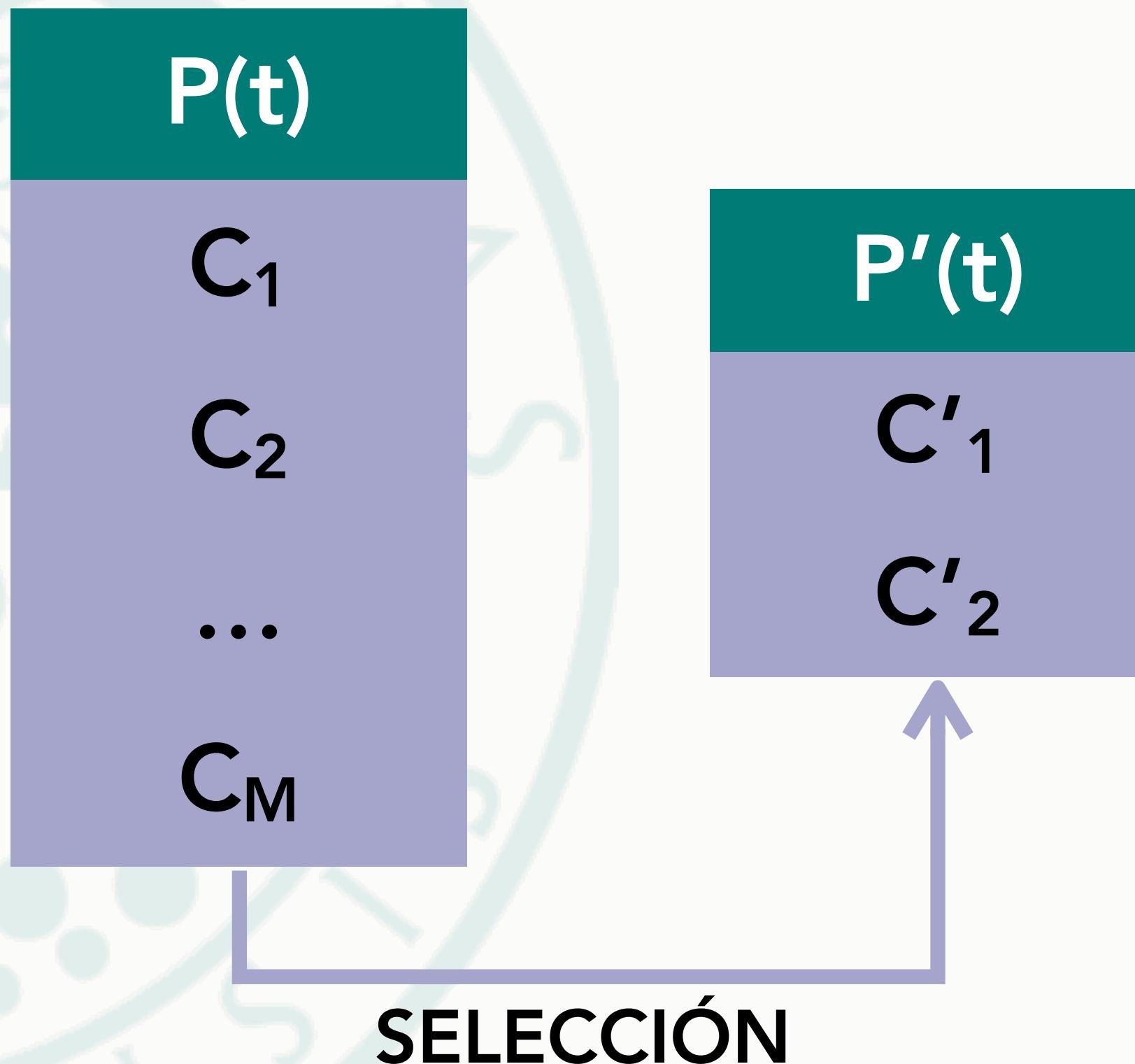
## Modelos evolutivos

modelo estacionario



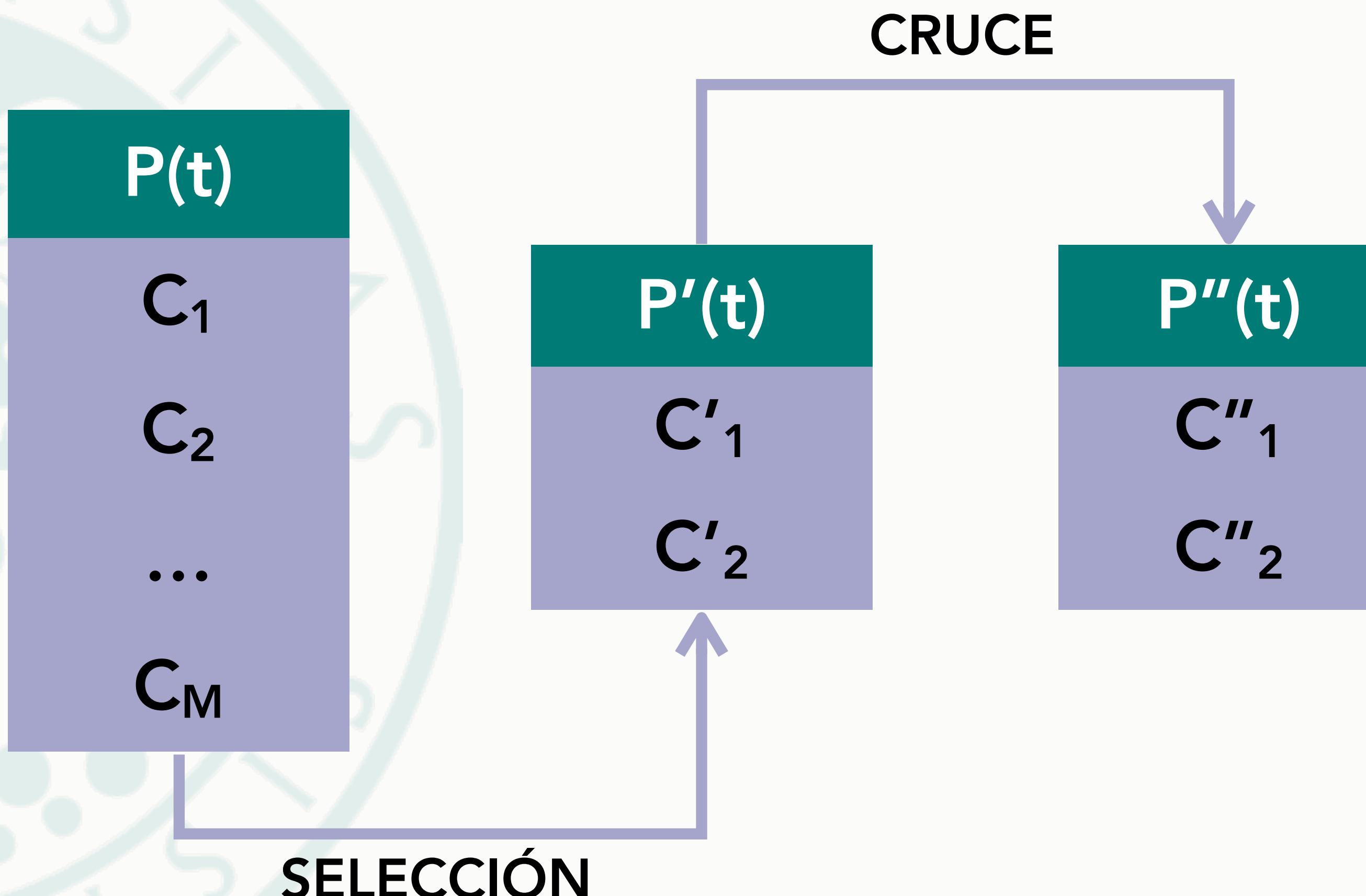
# Modelos evolutivos

modelo estacionario



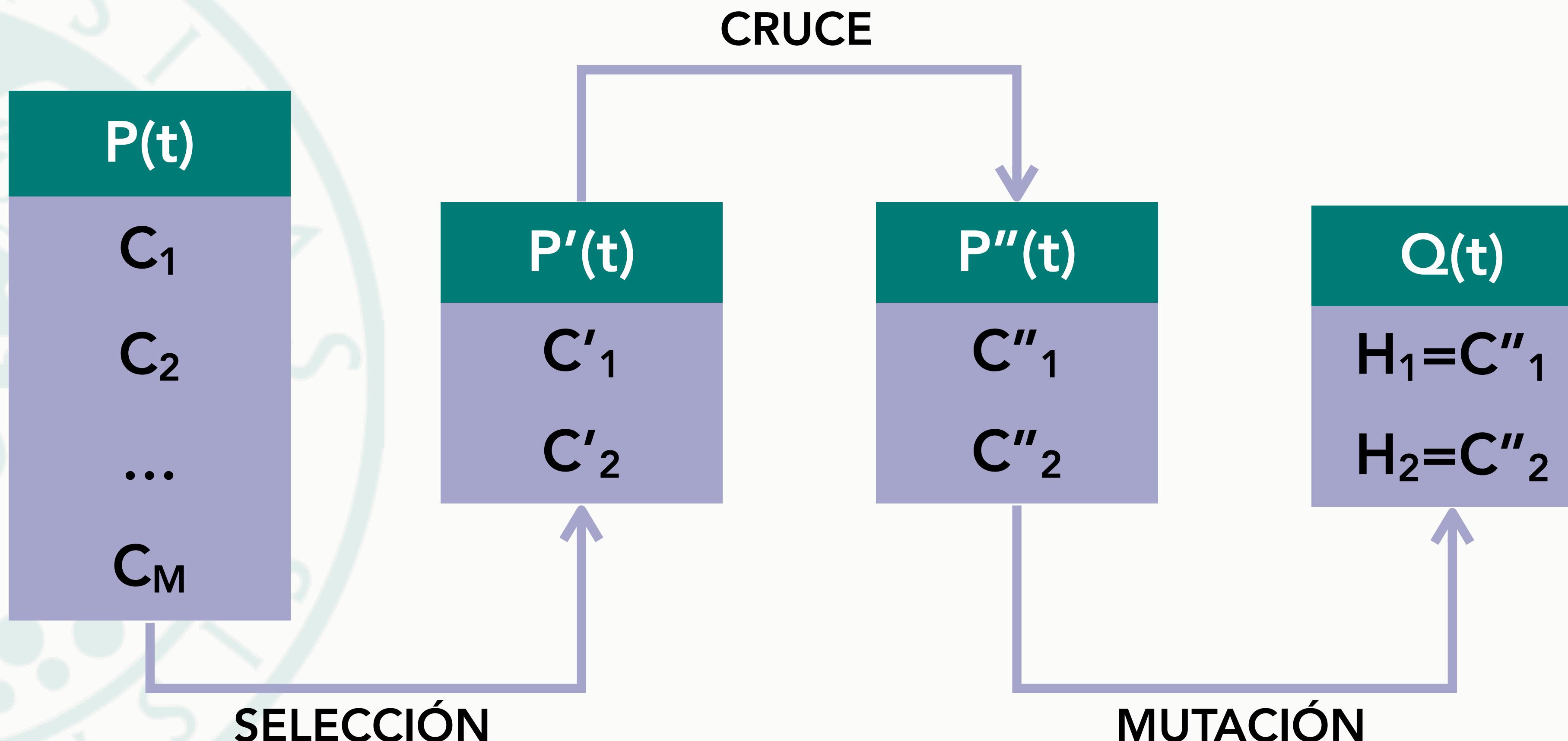
# Modelos evolutivos

modelo estacionario



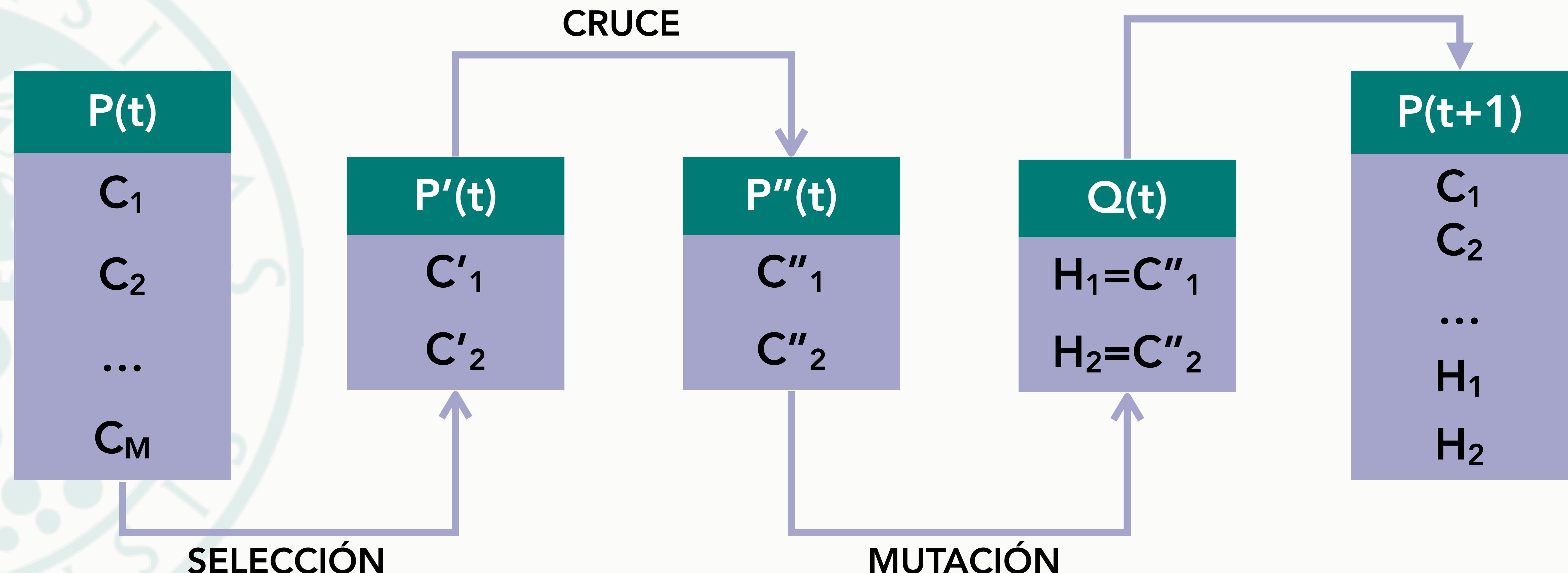
# Modelos evolutivos

modelo estacionario



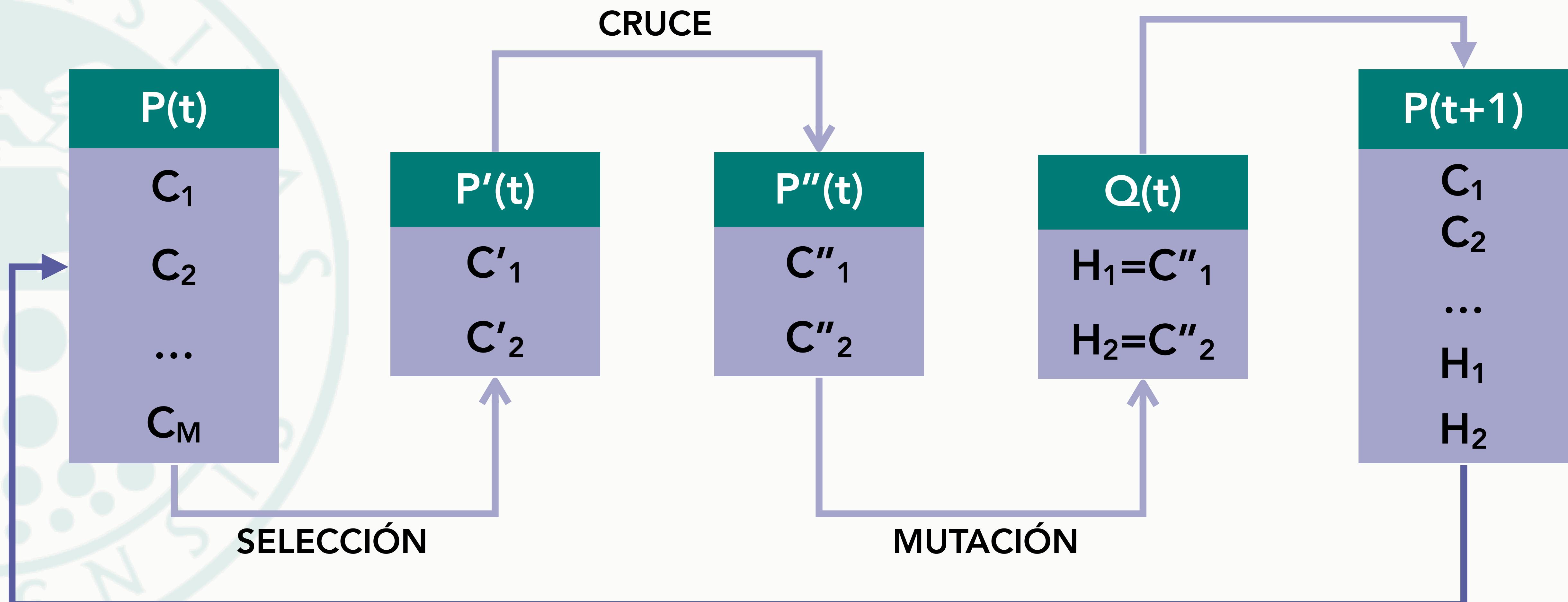
# Modelos evolutivos

modelo estacionario



# Modelos evolutivos

modelo estacionario



# Construcción de un algoritmo evolutivo

pasos necesarios



# Construcción de un algoritmo evolutivo

pasos necesarios

Dependen  
del  
problema

Componentes  
del  
algoritmo

Diseñar una representación

Decidir la forma de iniciar la población

Diseñar una correspondencia entre genotipo y fenotipo

Diseñar de una forma de evaluar un individuo

Diseñar un operador de cruce

Diseñar un operador de mutación

Decidir la selección de los individuos

Decidir el reemplazamiento de los individuos

Decidir la condición de parada

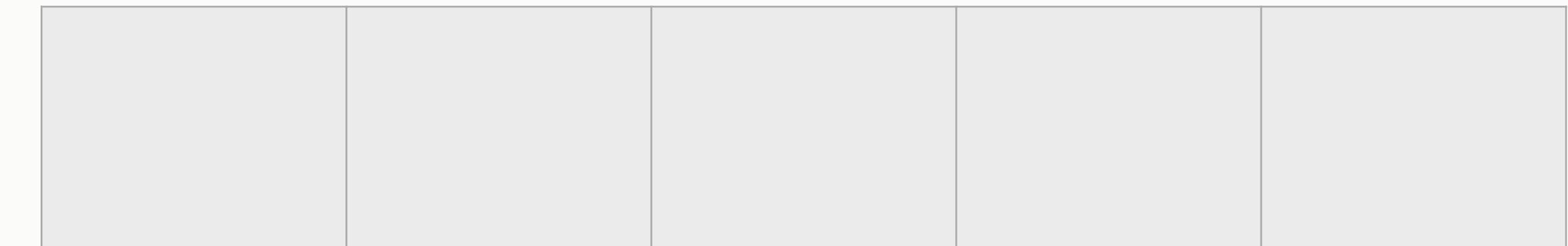


# Construcción de un algoritmo evolutivo

representación

- Mecanismo de codificación del genotipo
- Elegir la más adecuada para el problema
- Considerar la evaluación posterior de este genotipo

**GENOTIPO**



- BINARIA
- ENTERA
- REAL
- SECUENCIA
- ORDEN

# Construcción de un algoritmo evolutivo

representación

**GENOTIPO (7 bits)**

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

**real**

-1.2	3.5	0	2.75	-3.2	9.8	1
------	-----	---	------	------	-----	---

**lista**

3	3	2	5	1
---	---	---	---	---

**representación de orden**

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

# Construcción de un algoritmo evolutivo

## inicialización

- Debemos intentar que esté distribuida en el espacio de búsqueda
  - 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
- Incluir conocimiento experto

POBLACIÓN							
1	0	0	1	1	1	0	
0	1	0	0	0	1	0	0
1	1	1	0	1	0	0	
0	0	1	0	1	0	0	
0	1	0	0	0	0	1	0
0	1	0	1	0	1	0	
1	0	1	0	1	0	0	
0	1	0	1	0	0	0	1

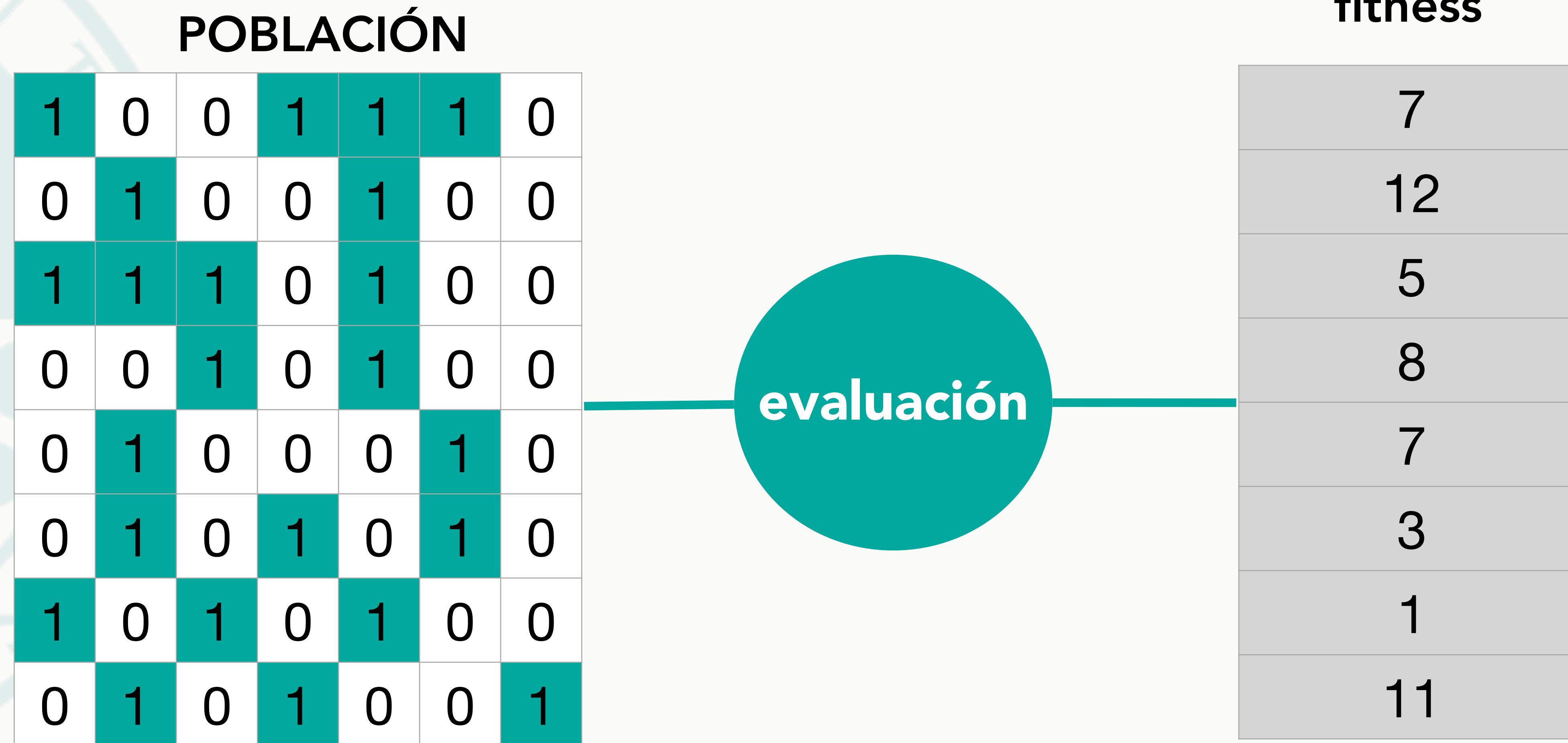
# Construcción de un algoritmo evolutivo

## evaluación

- Es el paso habitualmente más costoso
- Puede ser una subrutina, un simulador, o cualquier proceso externo
- 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

## Construcción de un algoritmo evolutivo

evaluación



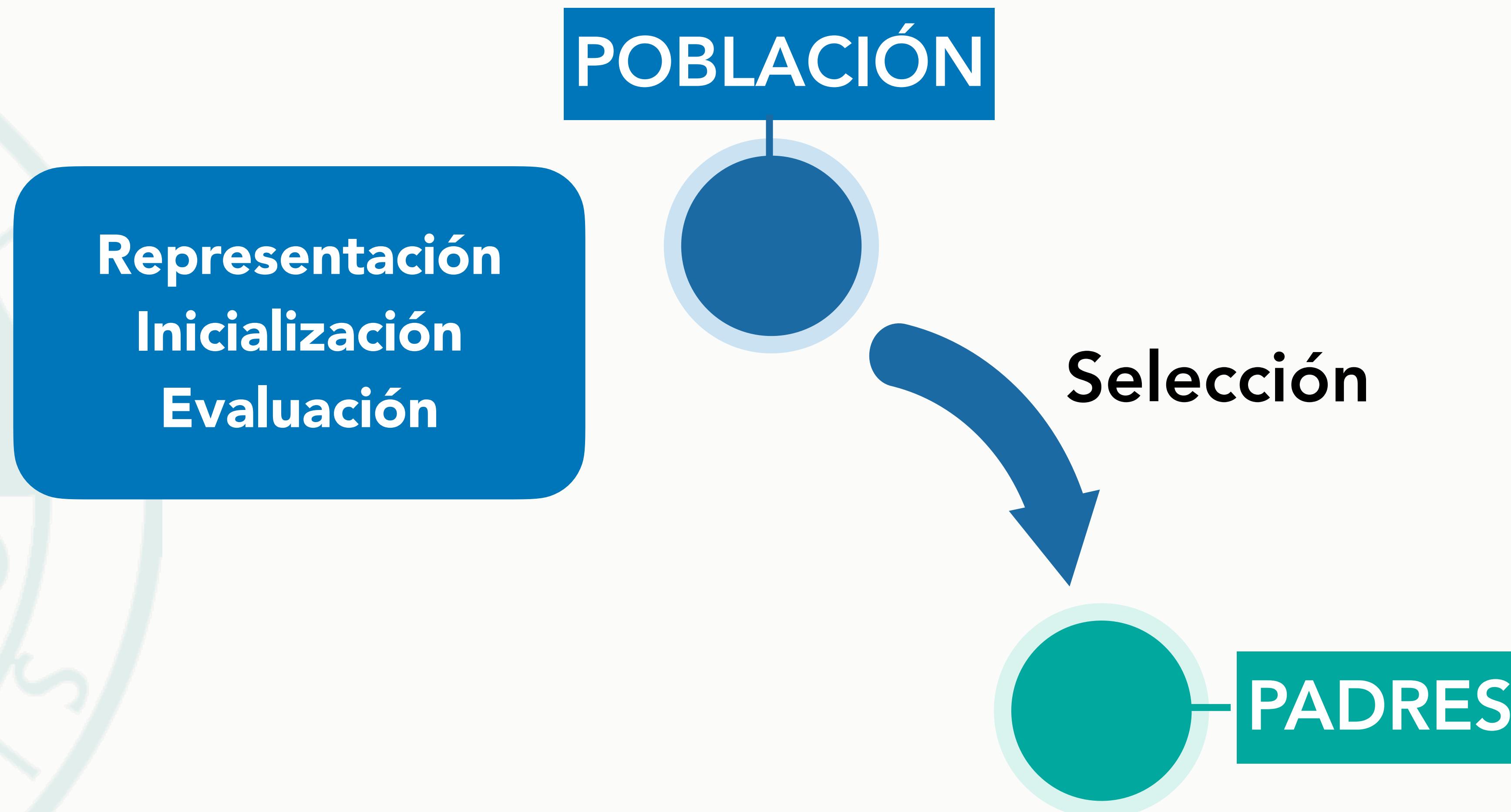
# Construcción de un algoritmo evolutivo

evaluación

POBLACIÓN

1	0	0	1	1	1	0	7
0	1	0	0	1	0	0	12
1	1	1	0	1	0	0	5
0	0	1	0	1	0	0	8
0	1	0	0	0	1	0	7
0	1	0	1	0	1	0	3
1	0	1	0	1	0	0	1
0	1	0	1	0	0	1	11





# Construcción de un algoritmo evolutivo

## selección

- Se debe garantizar que los **mejores individuos tengan una mayor posibilidad de ser padres**, es decir, reproducirse frente a los individuos peor adaptados
- Hay que tener especial cuidado para **dar oportunidad también a los menos buenos**. Éstos individuos pueden incluir material genético útil en el proceso de reproducción
- Esta idea define la **presión selectiva** que determina en qué grado la reproducción está dirigida por los mejores individuos

# Construcción de un algoritmo evolutivo

## selección

- El proceso de selección se basa principalmente en la función de adaptación o *fitness*
- Esquemas de selección:
  - **Aleatorio**: Completamente aleatoria
  - **Ruleta**: Se asigna una probabilidad de selección proporcional al valor del fitness
  - **Torneo**: Escoge al individuo de mejor fitness entre seleccionados aleatoriamente

# Construcción de un algoritmo evolutivo

selección por ruleta

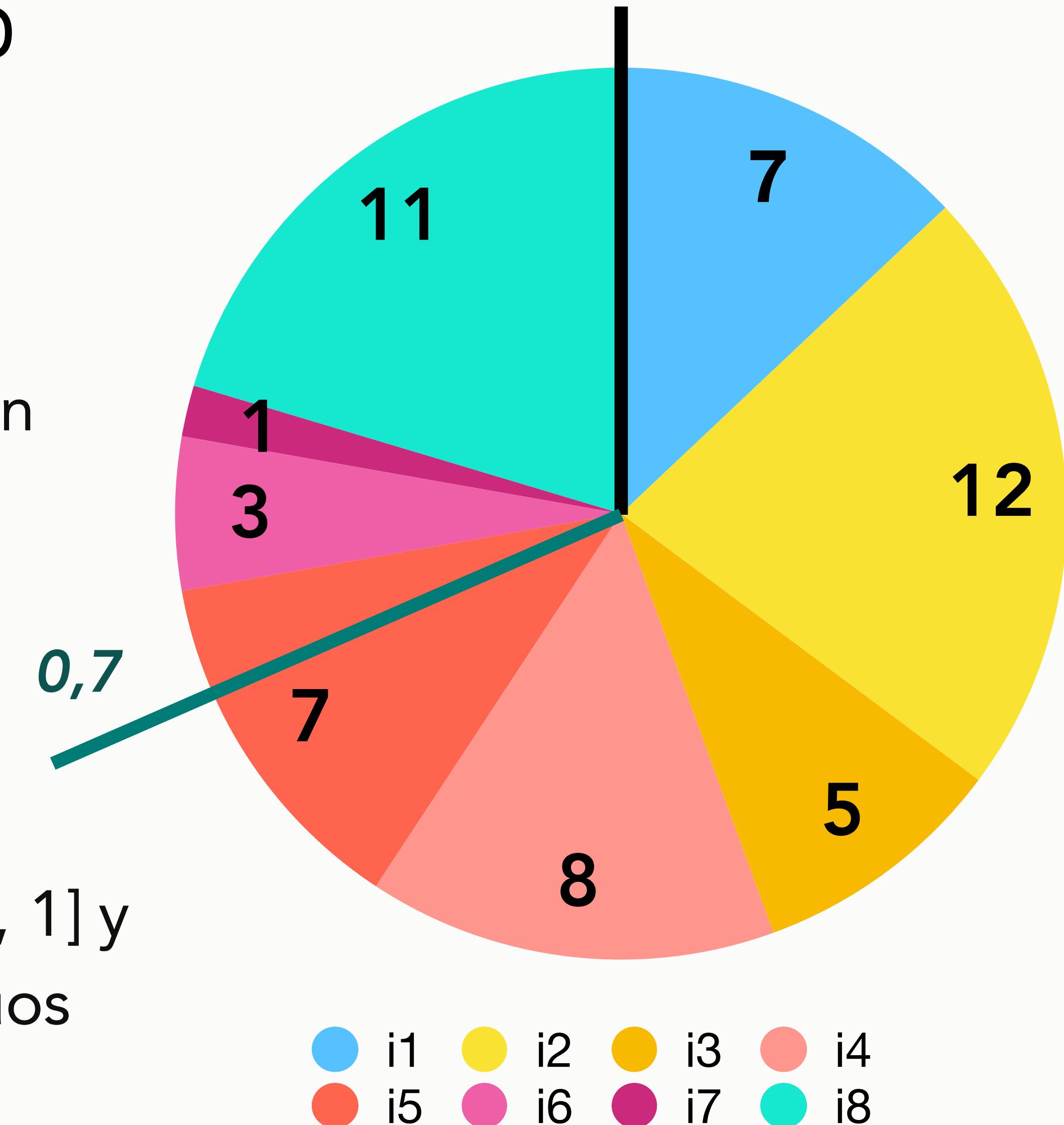
- Es el algoritmo tradicional
- Se asigna una probabilidad de selección proporcional al fitness
- Se gira la ruleta tantas veces como individuos seleccionemos
- Se genera un número en el intervalo  $[0, 1]$  y se va acumulando para extraer individuos
- Presión selectiva muy alta



# Construcción de un algoritmo evolutivo

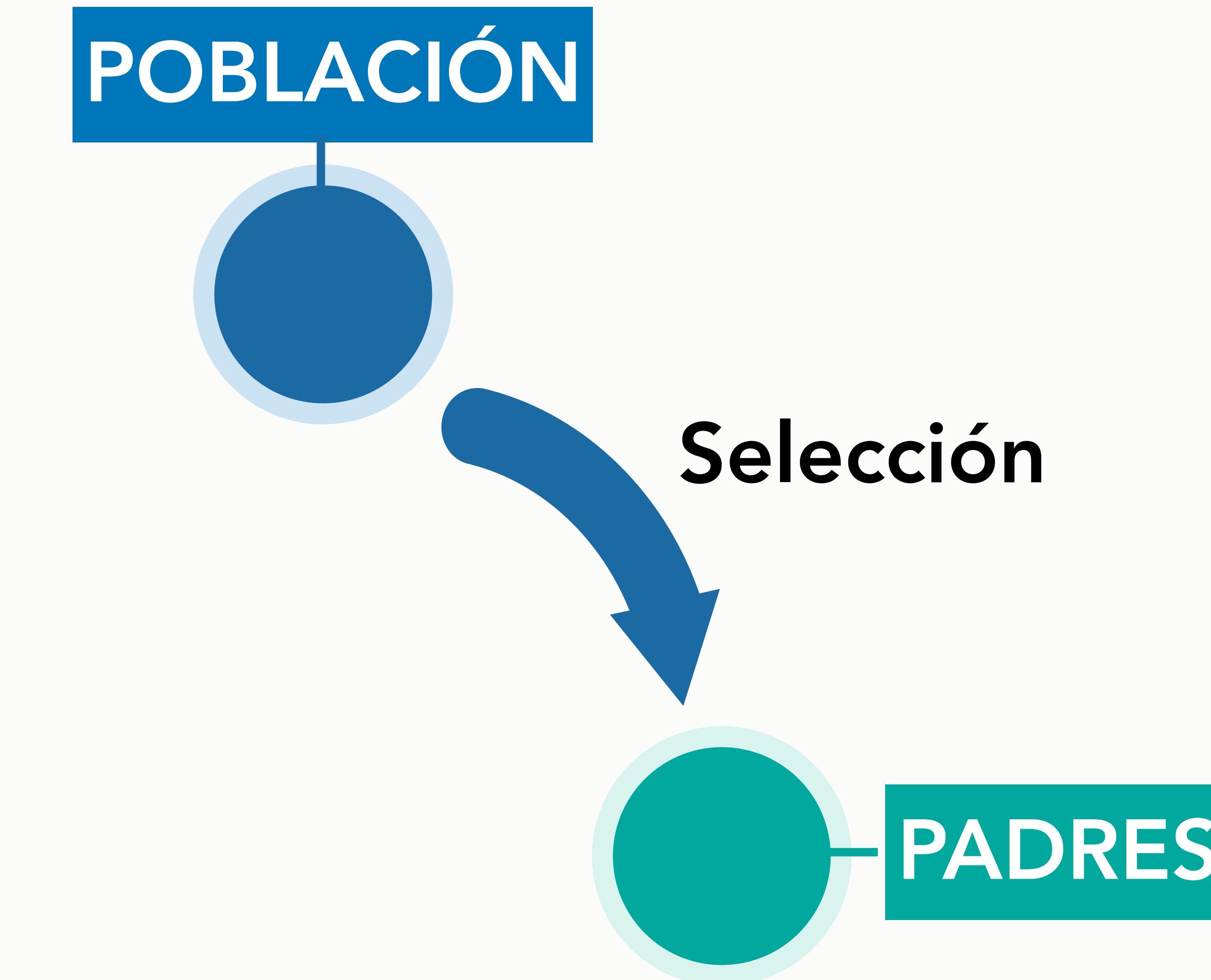
selección por ruleta

- Es el algoritmo tradicional
- Se asigna una probabilidad de selección proporcional al fitness
- Se gira la ruleta tantas veces como individuos seleccionemos
- Se genera un número en el intervalo  $[0, 1]$  y se va acumulando para extraer individuos
- Presión selectiva muy alta



# Modelo evolutivo

editar subtítulo



# Modelo evolutivo

editar subtítulo

DESCENDIENTES

POBLACIÓN

Selección

PADRES

Operadores  
genéticos

# Construcción de un algoritmo evolutivo

## operador de cruce

- Aspectos importantes en un cruce:
  - Los hijos deberían heredar algunas características de cada padre
  - Se debe diseñar de acuerdo a la representación
  - La recombinación debe producir cromosomas válidos
  - Tiene una probabilidad  $P_c$  alta de actuación sobre cada pareja de padres a cruzar, entre un 60-90%
  - En un problema se pueden implementar distintos operadores de cruce

# Construcción de un algoritmo evolutivo

operador de cruce

1	0	0	1	1	1	0	7
0	0	1	0	1	0	0	8

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

Operador de cruce en un punto

# Construcción de un algoritmo evolutivo

operador de cruce para representación real

Para este tipo de representación existen muchos operadores. En este caso vamos a ver una recombinación aritmética

2	0.5	1.3	0.1	1	1.2	3.2	7
0.5	1.2	0.5	0	1	0.6	2.9	8

(2+0.5)/2	(0.5+1.2)/2	(1.3+0.5)/2	(0.1+0)/2	(1+1)/2	(1.2+0.6)/2	(3.2+2.9)/2	-
-----------	-------------	-------------	-----------	---------	-------------	-------------	---

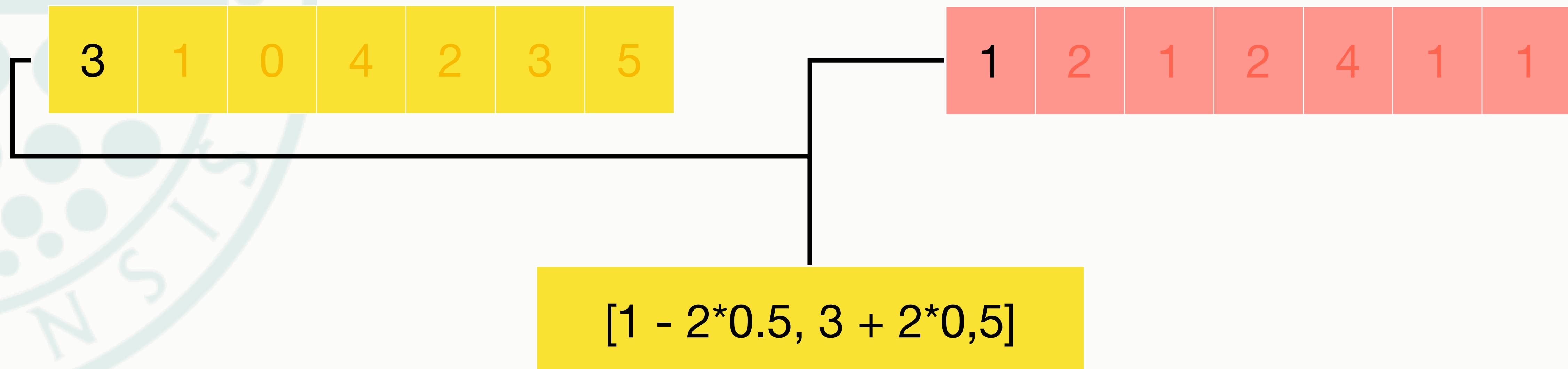
1.25	0.85	0.9	0.05	1	0.9	3.05	-
------	------	-----	------	---	-----	------	---

# Construcción de un algoritmo evolutivo

operador de cruce BLX-alfa

- Considerando un alfa en el intervalo  $[0, 1]$ , por ejemplo: 0.5

Cada alelo se genera aleatoriamente en el intervalo:  
 $[\min - \text{Int} * \text{alfa}, \max + \text{Int} * \text{alfa}]$



# Construcción de un algoritmo evolutivo

## operador de mutación

- Se pueden tener uno o más operadores de mutación para la representación
- Algunos aspectos importantes:
  - Debe permitir alcanzar cualquier parte del espacio de búsqueda
  - Se debe controlar el tamaño de la mutación
  - Debe producir cromosomas válidos
  - Se aplica con probabilidad baja sobre cada descendiente tras el cruce

# Construcción de un algoritmo evolutivo

## operador de mutación

- La mutación ocurre con una probabilidad  $P_m$  para cada gen
- Aquí podemos ver mutación aleatoria para representación binaria y entera

**BINARIA**

1	0	0	1	1	1	0	7
---	---	---	---	---	---	---	---

**ENTERA**

3	1	5	1	3	2	0	3
---	---	---	---	---	---	---	---

**Mutación de 1 bit**

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

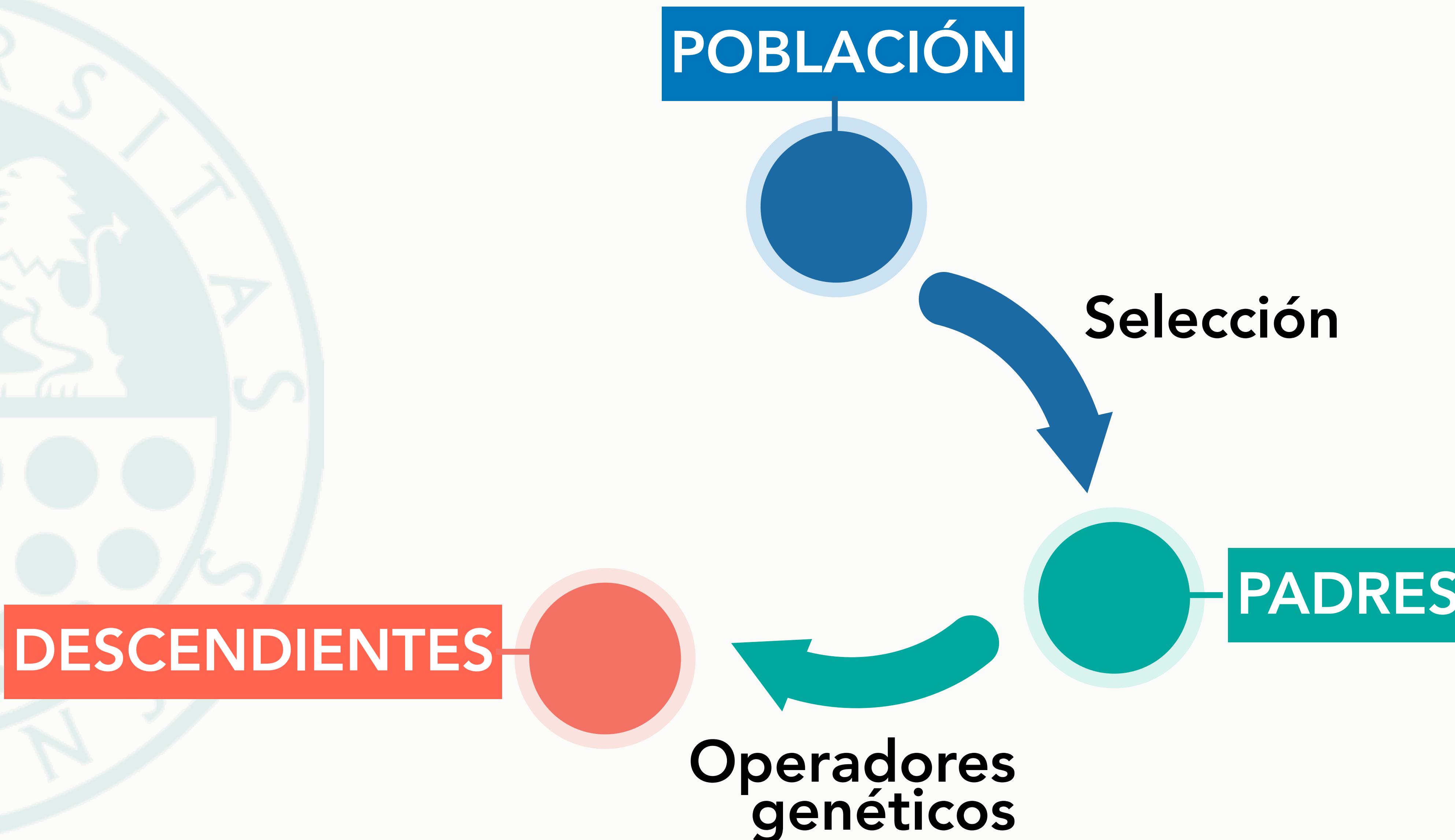
3	2	5	1	3	2	0	-
---	---	---	---	---	---	---	---

**Mutación múltiple**

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

3	1	1	2	1	2	0	-
---	---	---	---	---	---	---	---

# Modelo evolutivo



# Modelo evolutivo



# Construcción de un algoritmo evolutivo

reemplazamiento

Operador relevante para conseguir una buena exploración y explotación

- La presión selectiva se ve también afectada por la forma en que los cromosomas de la población son reemplazados por los nuevos descendientes
- Métodos aleatorios o determinísticos
- **Elitismo** >> Aconsejado en **modelos generacionales** para no perder la mejor solución
  - Alto elitismo consiste en utilizar una población intermedia con todos los padres ( $N$ ) y todos los descendientes y seleccionar los  $N$  mejores. Esto se combina con componentes de alto grado de diversidad

# Construcción de un algoritmo evolutivo

reemplazamiento para algoritmos estacionarios

- En un modelo estacionario existen distintas propuestas:
  - **Reemplazar al peor** >> Genera alta presión selectiva
  - **Torneo restringido** >> Reemplaza el más parecido de entre un número w. Mantiene una cierta diversidad
  - **Peor entre semejantes** >> Se reemplaza el peor del conjunto de los w padres más parecidos al descendiente generado. Equilibrio entre diversidad y presión selectiva
  - **Algoritmo de crowding determinístico** >> El hijo reemplaza a su parente más parecido. Mantiene diversidad

# Construcción de un algoritmo evolutivo

condición de parada

- Cuando se alcanza el óptimo o una solución con un nivel de calidad aceptable
- Recursos limitados de la CPU como el número de evaluaciones o el número de generaciones
- Límite sobre la paciencia del usuario como después de algunas iteraciones sin mejora

## Algoritmo evolutivo básico para maximizar una función matemática

Diseña un algoritmo evolutivo que devuelva el número entero que maximiza la siguiente función matemática:

$$f(x) = x^2, \text{ en el rango } 0-63 \text{ para enteros}$$

La representación de los cromosomas se debe realizar mediante código binario

# Algoritmo evolutivo básico para maximizar una función matemática



# Algoritmo evolutivo básico para maximizar una función matemática

## REPRESENTACIÓN

Representación binaria de un número por debajo de 64 necesitamos 6 bits

### GENOTIPO

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

$$2^5 + 2^2 + 2^1 = 38$$

### FENOTIPO

# Algoritmo evolutivo básico para maximizar una función matemática

## REPRESENTACIÓN

Representación binaria de un número por debajo de 64 necesitamos 6 bits

### GENOTIPO

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

$$2^5 + 2^2 + 2^1 = 38$$

### FENOTIPO

## EVALUACIÓN

1	0	0	1	1	0	$f(x)=1444$
---	---	---	---	---	---	-------------

# Algoritmo evolutivo básico para maximizar una función matemática

## REPRESENTACIÓN

Representación binaria de un número por debajo de 64 necesitamos 6 bits

### GENOTIPO

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

$$2^5 + 2^2 + 2^1 = 38$$

### FENOTIPO

## INICIALIZACIÓN

1	0	0	1	1	0
1	1	0	0	0	0
0	0	1	0	0	0
0	1	0	1	1	0

## EVALUACIÓN

1	0	0	1	1	0	$f(x)=1444$
---	---	---	---	---	---	-------------

# Algoritmo evolutivo básico para maximizar una función matemática

## INICIALIZACIÓN

1	0	0	1	1	0
1	1	0	0	0	0
0	0	1	0	0	0
0	1	0	1	1	0

## EVALUACIÓN

1	0	0	1	1	0
$f(x)=1444$					

# Algoritmo evolutivo básico para maximizar una función matemática

## INICIALIZACIÓN

1	0	0	1	1	0
1	1	0	0	0	0
0	0	1	0	0	0
0	1	0	1	1	0

## FENOTIPO $f(x)$

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
0	0	1	0	0	0	8	64
0	1	0	1	1	0	22	484

## EVALUACIÓN

1	0	0	1	1	0	$f(x)=1444$
---	---	---	---	---	---	-------------

# Algoritmo evolutivo básico para maximizar una función matemática

## INICIALIZACIÓN

1	0	0	1	1	0
1	1	0	0	0	0
0	0	1	0	0	0
0	1	0	1	1	0

## FENOTIPO $f(x)$

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
0	0	1	0	0	0	8	64
0	1	0	1	1	0	22	484

## SELECCIÓN POR RULETA

## EVALUACIÓN

1	0	0	1	1	0	$f(x)=1444$
---	---	---	---	---	---	-------------

# Algoritmo evolutivo básico para maximizar una función matemática

## INICIALIZACIÓN

1	0	0	1	1	0
1	1	0	0	0	0
0	0	1	0	0	0
0	1	0	1	1	0

## FENOTIPO $f(x)$

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
0	0	1	0	0	0	8	64
0	1	0	1	1	0	22	484

## SELECCIÓN POR RULETA

1	0	0	1	1	0	38	1444
---	---	---	---	---	---	----	------

## EVALUACIÓN

1	0	0	1	1	0	$f(x)=1444$
---	---	---	---	---	---	-------------

# Algoritmo evolutivo básico para maximizar una función matemática

## INICIALIZACIÓN

1	0	0	1	1	0
1	1	0	0	0	0
0	0	1	0	0	0
0	1	0	1	1	0

## FENOTIPO $f(x)$

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
0	0	1	0	0	0	8	64
0	1	0	1	1	0	22	484

## SELECCIÓN POR RULETA

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304

## EVALUACIÓN

1	0	0	1	1	0	$f(x)=1444$
---	---	---	---	---	---	-------------

# Algoritmo evolutivo básico para maximizar una función matemática

## INICIALIZACIÓN

1	0	0	1	1	0
1	1	0	0	0	0
0	0	1	0	0	0
0	1	0	1	1	0

## FENOTIPO $f(x)$

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
0	0	1	0	0	0	8	64
0	1	0	1	1	0	22	484

## SELECCIÓN POR RULETA

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
1	1	0	0	0	0	48	2304

## EVALUACIÓN

1	0	0	1	1	0	$f(x)=1444$
---	---	---	---	---	---	-------------

# Algoritmo evolutivo básico para maximizar una función matemática

## INICIALIZACIÓN

1	0	0	1	1	0
1	1	0	0	0	0
0	0	1	0	0	0
0	1	0	1	1	0

## FENOTIPO $f(x)$

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
0	0	1	0	0	0	8	64
0	1	0	1	1	0	22	484

## SELECCIÓN POR RULETA

## EVALUACIÓN

1	0	0	1	1	0	$f(x)=1444$
---	---	---	---	---	---	-------------

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
1	1	0	0	0	0	48	2304
0	1	0	1	1	0	22	484

# Algoritmo evolutivo básico para maximizar una función matemática

						FENOTIPO	$f(x)$
1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
0	0	1	0	0	0	8	64
0	1	0	1	1	0	22	484

						PADRES	
1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
1	1	0	0	0	0	48	2304
0	1	0	1	1	0	22	484

# Algoritmo evolutivo básico para maximizar una función matemática

**FENOTIPO f(x)**

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
0	0	1	0	0	0	8	64
0	1	0	1	1	0	22	484

**HIJOS**

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
1	1	0	0	0	0	48	2304
0	1	0	1	1	0	22	484

**PADRES**

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
1	1	0	0	0	0	48	2304
0	1	0	1	1	0	22	484

# Algoritmo evolutivo básico para maximizar una función matemática

**FENOTIPO f(x)**

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
0	0	1	0	0	0	8	64
0	1	0	1	1	0	22	484

**HIJOS**

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
1	1	0	0	0	0	48	2304
0	1	0	1	1	0	22	484

**PADRES**

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
1	1	0	0	0	0	48	2304
0	1	0	1	1	0	22	484

1	1	0	1	1	0	54	2916
1	0	0	0	0	0	32	1024
1	1	1	0	1	0	58	3364
0	1	0	1	1	0	20	400

# Algoritmo evolutivo básico para maximizar una función matemática

## NUEVA POBLACIÓN DEL SIGUIENTE CICLO

1	1	0	1	1	0	54	2916
1	0	0	0	0	0	32	1024
1	1	1	0	1	0	58	3364
0	1	0	1	0	0	20	400

# Algoritmo evolutivo básico para maximizar una función matemática

**FENOTIPO f(x)**

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
0	0	1	0	0	0	8	64
0	1	0	1	1	0	22	484

**HIJOS**

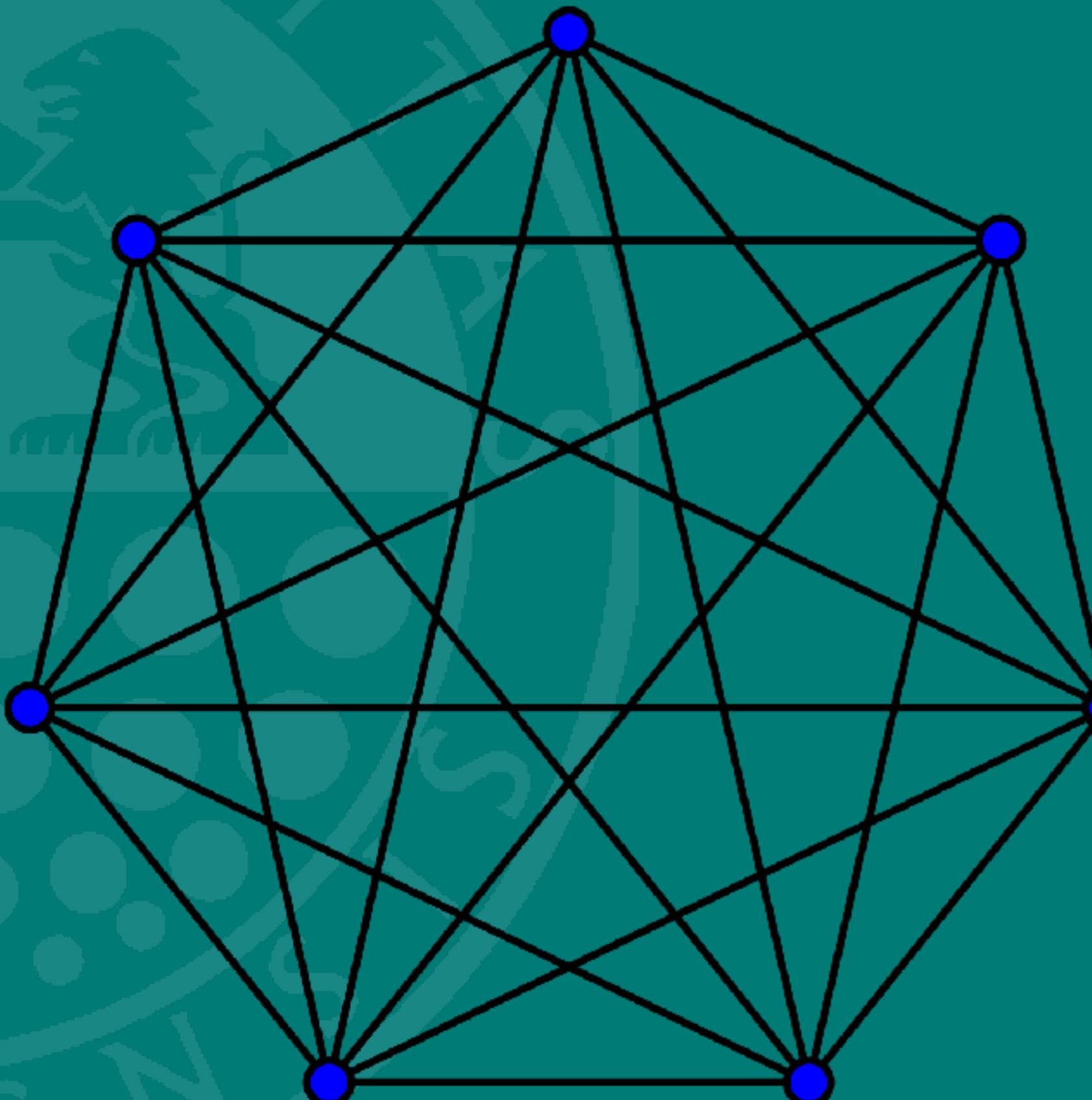
1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
1	1	0	0	0	0	48	2304
0	1	0	1	1	0	22	484

**PADRES**

1	0	0	1	1	0	38	1444
1	1	0	0	0	0	48	2304
1	1	0	0	0	0	48	2304
0	1	0	1	1	0	22	484

1	1	0	1	1	0	54	2916
1	0	0	0	0	0	32	1024
1	1	1	0	0	0	58	3364
0	1	0	1	1	0	20	400

# El viajante de comercio

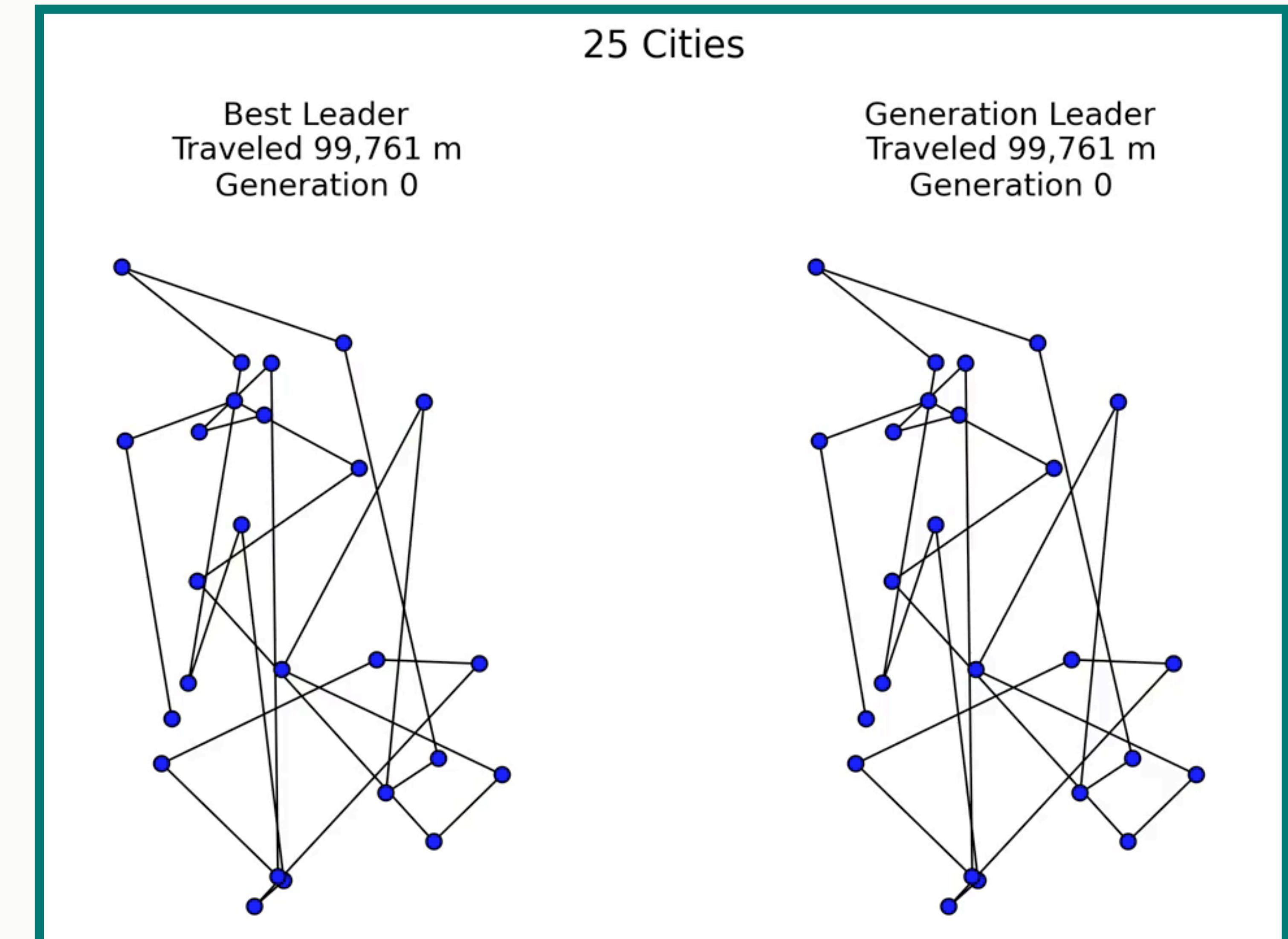


- Problema con 7 ciudades
- Objetivo: Suma de la distancia entre las ciudades
- Solo se pasa por la ciudad, una única vez
- Posibles combinaciones  $7! = 5040$
- Posibles combinaciones  $25! = 1.55e25$

# El viajante de comercio

- Representación de orden
- Inicialización aleatoria
- Población de 60 individuos
- Selección por ruleta
- Permuta de dos ciudades

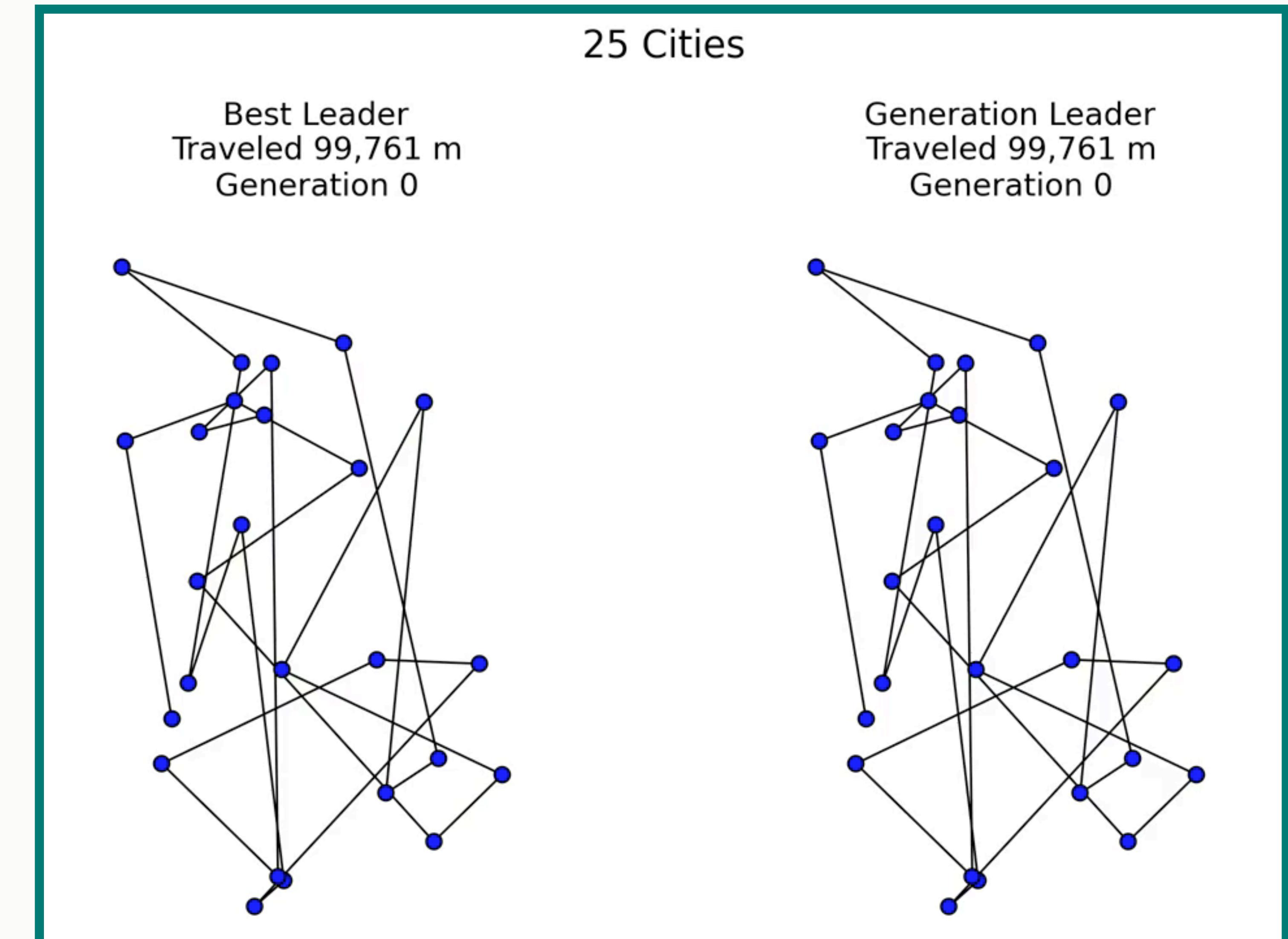
[youtube.com/watch?v=efTyb82GEDw&t=1s](https://youtube.com/watch?v=efTyb82GEDw&t=1s)



## El viajante de comercio

- Representación de orden
- Inicialización aleatoria
- Población de 60 individuos
- Selección por ruleta
- Permuta de dos ciudades

[youtube.com/watch?v=efTyb82GEDw&t=1s](https://youtube.com/watch?v=efTyb82GEDw&t=1s)



# Algoritmo de optimización de corte de bananas



*Diseña un algoritmo evolutivo que una vez recibido un pedido de una empresa obtenga una receta de corte que reduzca el desperdicio*

# Algoritmo de optimización de corte de bananas



# Algoritmo de optimización de corte de bananas



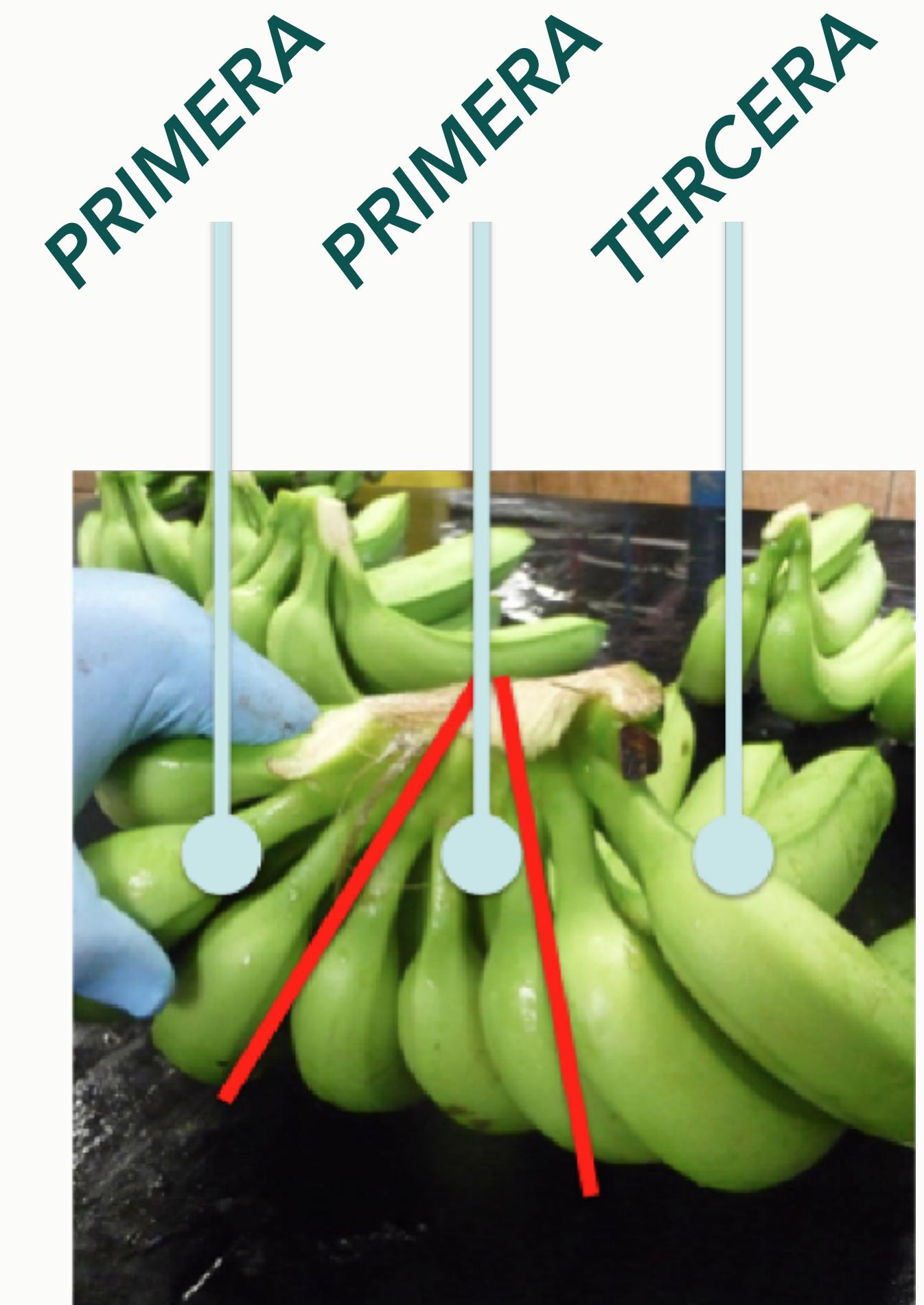
# Algoritmo de optimización de corte de bananas

- Tenemos un conjunto de racimos y un pedido
  - Primera calidad: 3000 kg
  - Segunda calidad: 1500 kg
  - Tercera calidad: 500 kg
- La empresa necesita empaquetar un pedido
- OBJETIVO: Determinar para ese pedido el orden en que deben aplicarse los cortes sobre cada racimo de bananas para minimizar el desperdicio



# Algoritmo de optimización de corte de bananas

- Tenemos un conjunto de racimos y un pedido
  - Primera calidad: 3000 kg
  - Segunda calidad: 1500 kg
  - Tercera calidad: 500 kg
- La empresa necesita empaquetar un pedido
- OBJETIVO: Determinar para ese pedido el orden en que deben aplicarse los cortes sobre cada racimo de bananas para minimizar el desperdicio



# Algoritmo de optimización de corte de bananas

## Metáfora

## Optimización

Evolución

Empaquetar el pedido por tipos

Individuo

Orden de los tipos de corte a aplicar

*Fitness - REDUCIR*

Peso de las bananas desperdiciadas

Gen

Tipo de corte

# Algoritmo de optimización de corte de bananas

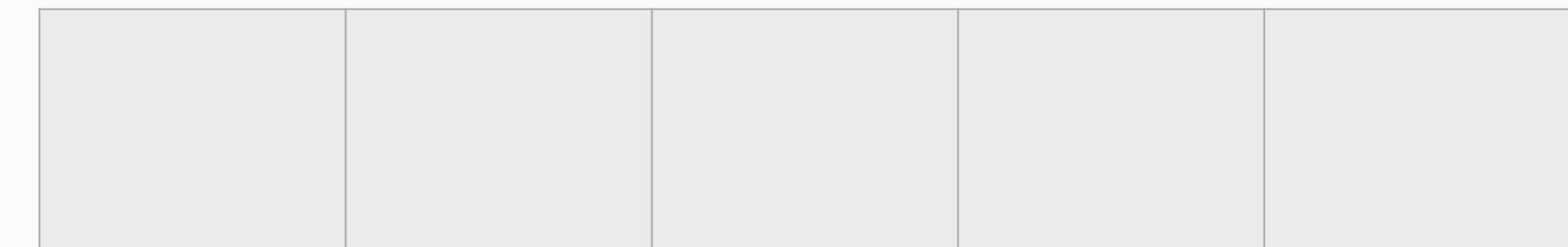
representación

Pedido	
Primera	3000
Segunda	1500
Tercera	500

**Receta  
de  
corte**

- Primera
- Tercera
- Segunda
- Primera
- Segunda

**GENOTIPO (Tipos de corte)**



- BINARIA
- SECUENCIA
- REAL
- ENTERA

# Algoritmo de optimización de corte de bananas

representación

Pedido	
Primera	3000
Segunda	1500
Tercera	500

**Receta  
de  
corte**

- Primera
- Tercera
- Segunda
- Primera
- Segunda

**GENOTIPO (Tipos de corte)**

“01”	“11”	“10”	“01”	“10”
------	------	------	------	------

**BINARIA (2 bits)**

**Primera = 01**  
**Segunda = 10**  
**Tercera = 11**

- BINARIA
- SECUENCIA
- REAL
- ENTERA

# Algoritmo de optimización de corte de bananas

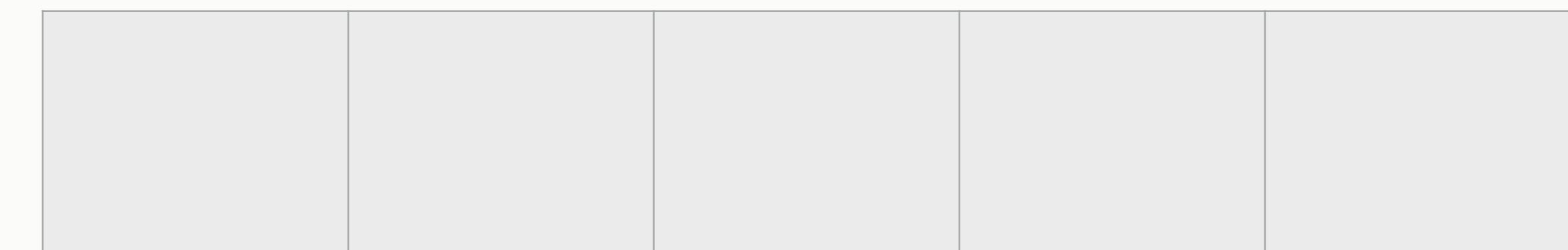
representación

	Pedido
Primera	3000
Segunda	1500
Tercera	500

**Receta  
de  
corte**

- Primera
- Tercera
- Segunda
- Primera
- Segunda

**GENOTIPO (Tipos de corte)**



**SECUENCIA**

**No tiene sentido porque un corte puede aparecer más de una vez**

- BINARIA
- SECUENCIA
- REAL
- ENTERA

# Algoritmo de optimización de corte de bananas

representación

Pedido	
Primera	3000
Segunda	1500
Tercera	500

**Receta  
de  
corte**

- Primera
- Tercera
- Segunda
- Primera
- Segunda

**GENOTIPO (Tipos de corte)**

--	--	--	--	--

**ENTERO**

**Primera = 1  
Segunda = 2  
Tercera = 3**

- BINARIA
- SECUENCIA
- REAL
- ENTERA

# Algoritmo de optimización de corte de bananas

representación

	Pedido
Primera	3000
Segunda	1500
Tercera	500

**Receta  
de  
corte**

- Primera
- Tercera
- Segunda
- Primera
- Segunda

**GENOTIPO (Tipos de corte)**

1	3	2	1	2
---	---	---	---	---

**ENTERO**

**Primera = 1**  
**Segunda = 2**  
**Tercera = 3**

- BINARIA
- SECUENCIA
- REAL
- ENTERA

# Algoritmo de optimización de corte de bananas

inicialización

- Podemos encontrar distintos mecanismos:
  - **Aleatoria por completo**
  - **Distribución probabilística**
  - Condicionado a una heurística
  - Por conocimiento experto

**POBLACIÓN**

1	3	2	2	1
2	3	2	1	1
3	3	3	2	1
3	3	2	2	3
1	1	3	2	3
1	2	3	1	3
1	1	3	2	2
1	3	2	3	1

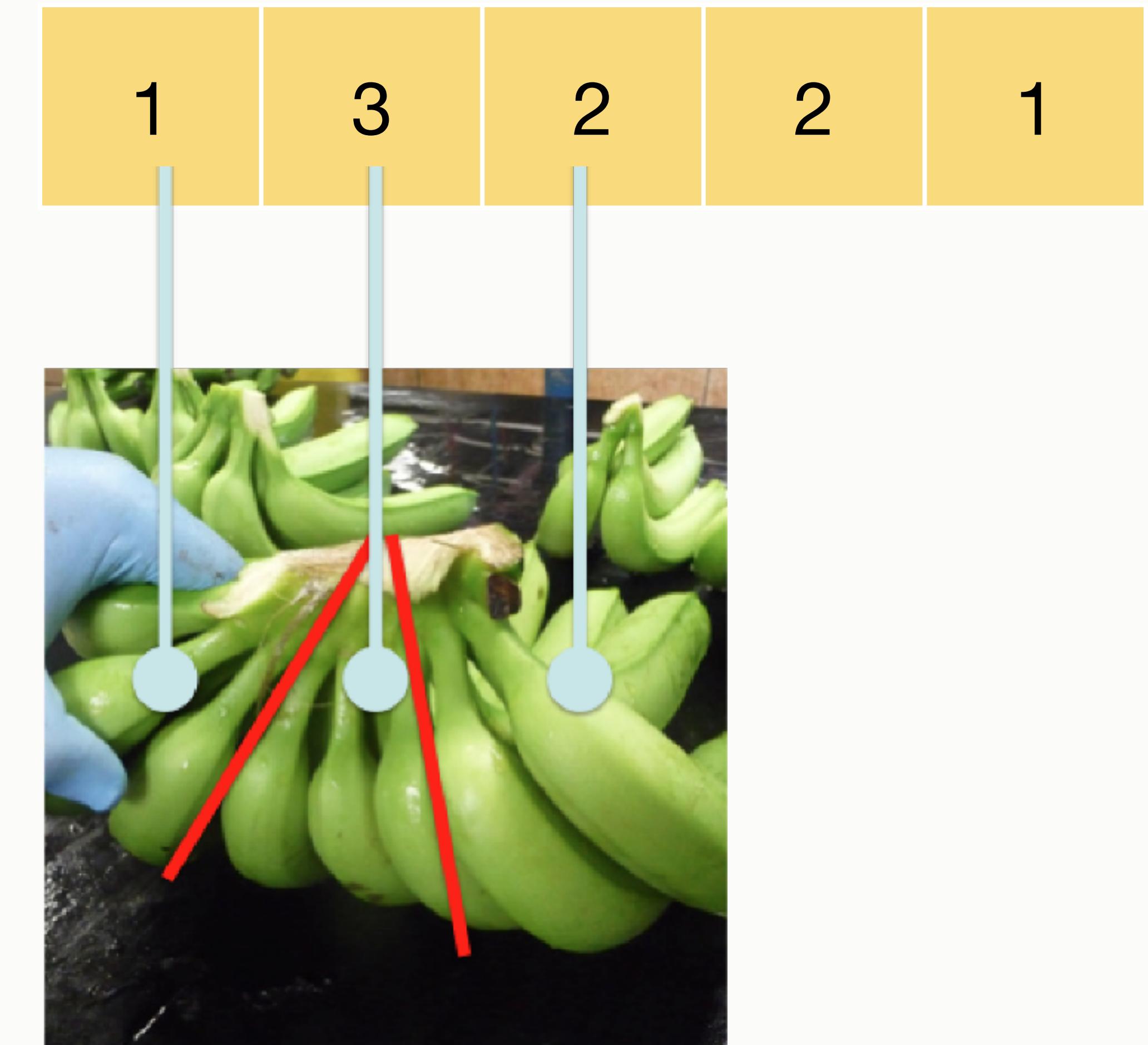
# Algoritmo de optimización de corte de bananas

evaluación



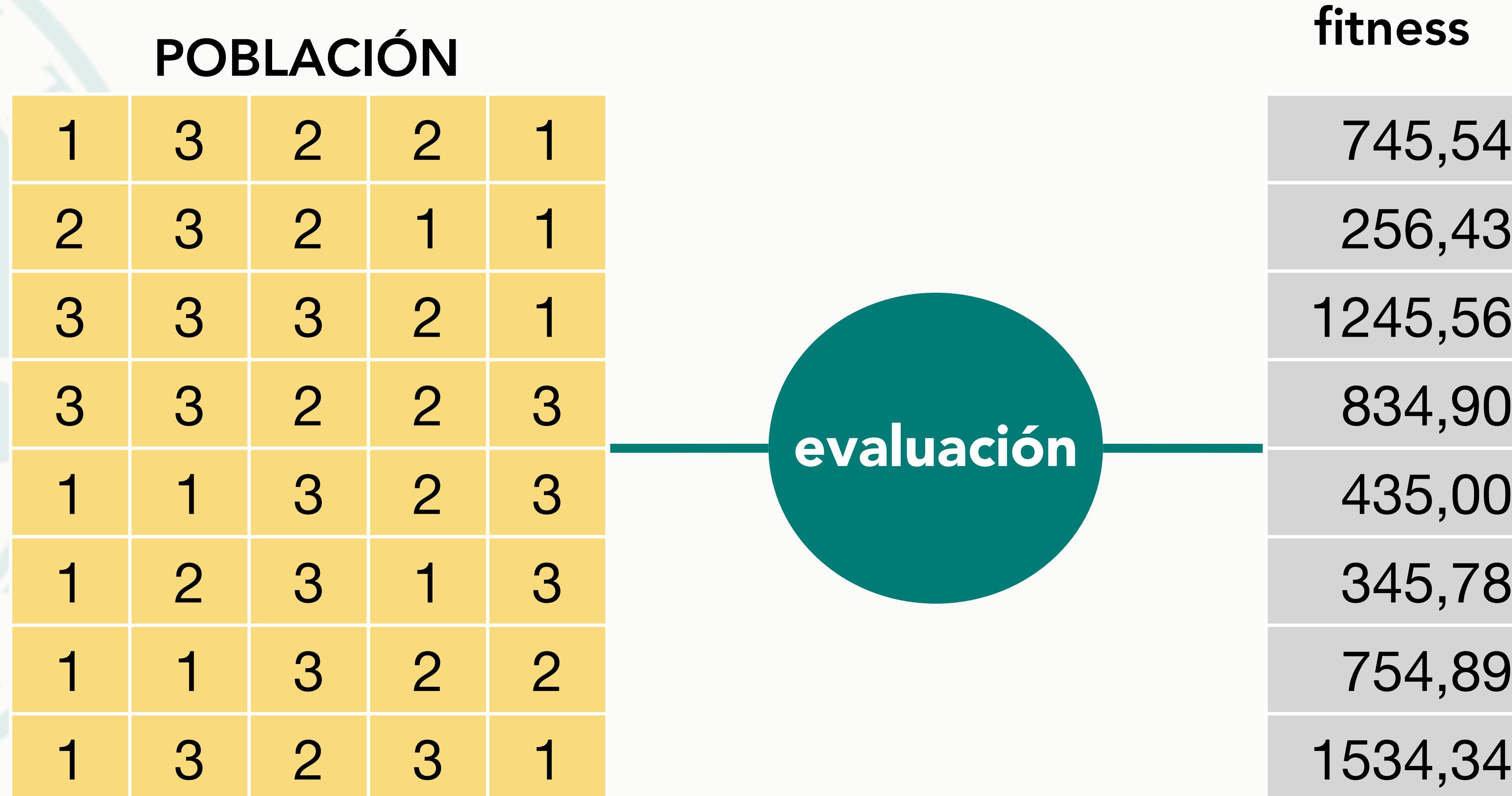
# Algoritmo de optimización de corte de bananas

evaluación



# Algoritmo de optimización de corte de bananas

evaluación



# Algoritmo de optimización de corte de bananas

evaluación

POBLACIÓN					fitness
1	3	2	2	1	745,54
2	3	2	1	1	256,43
3	3	3	2	1	1245,56
3	3	2	2	3	834,90
1	1	3	2	3	435,00
1	2	3	1	3	345,78
1	1	3	2	2	754,89
1	3	2	3	1	1534,34

# Algoritmo de optimización de corte de bananas

evaluación

POBLACIÓN					fitness
1	3	2	2	1	745,54
<b>2</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>1</b>	<b>256,43</b>
3	3	3	2	1	1245,56
3	3	2	2	3	834,90
1	1	3	2	3	435,00
1	2	3	1	3	345,78
1	1	3	2	2	754,89
1	3	2	3	1	1534,34

# Algoritmo de optimización de corte de bananas

selección por torneo

**Se escogen aleatoriamente k individuos**

1	3	2	2	1	745,54
2	3	2	1	1	256,43
3	3	3	2	1	1245,56
3	3	2	2	3	834,90
1	1	3	2	3	435,00
1	2	3	1	3	345,78
1	1	3	2	2	754,89
1	3	2	3	1	1534,34

1	3	2	2	1	745,54
1	1	3	2	3	435,00
1	1	3	2	2	754,89

**Torneo con k=3**

**Se coge el individuo con mejor fitness**

**Los torneos se repiten hasta completar la población nueva**

# Algoritmo de optimización de corte de bananas

operador de cruce y mutación

1	3	2	2	1	745,54
1	1	3	2	2	754,89

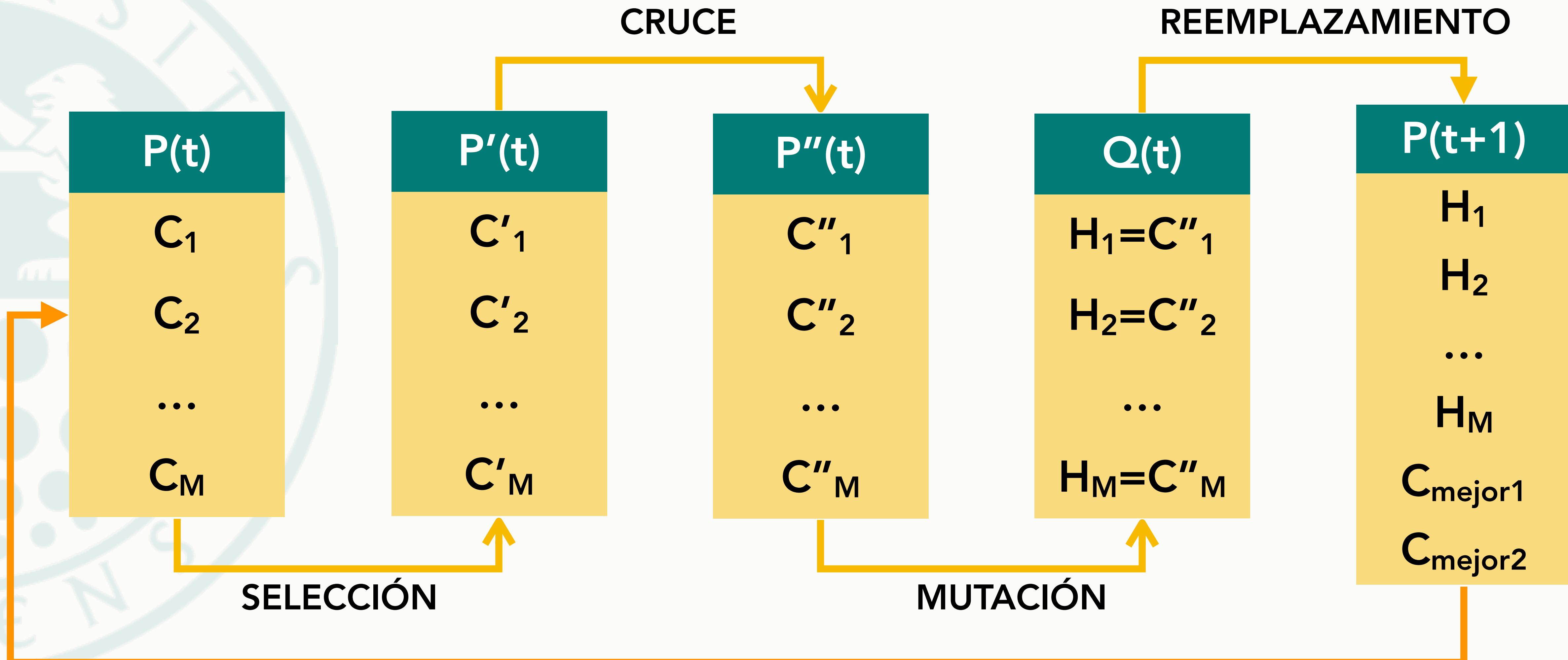
1	3	3	2	1	-
1	1	2	2	2	-

1	3	2	2	1	745,54
---	---	---	---	---	--------

1	2	2	2	1	-
---	---	---	---	---	---

# Algoritmo de optimización de corte de bananas

reemplazamiento generacional con élite de dos individuos



# Algoritmo de optimización de corte de bananas

condición de parada

**Un determinado número de generaciones**

# Algoritmo de optimización de corte de bananas

resultados en campo

- **Cortador en campo**
  - Desperdicios del 8%-12%
- **Algoritmo evolutivo**
  - Desperdicios del 5%-9%

*Cada punto de desperdicio equivalen a  
unos 100.000\$ por día*

# Algoritmo de optimización de corte de bananas

resultados en campo

- Cor

- ▶ Des

*Este ejemplo es una simplificación de un algoritmo evolutivo más complejo*

*Cada punto de desperdicio equivalen a unos 100.000\$ por día*

# Sobre la utilización de algoritmos evolutivos

- NUNCA sacar conclusiones de una única ejecución:
  - Utilizar medidas estadísticas como medias, medianas, etc
  - Deben realizarse un número suficiente de ejecuciones independientes
- No ajustar/comprobar un algoritmo sobre ejemplos sencillos si queremos trabajar con casos reales

## Resumen

- Basados en una metáfora biológica: evolución
- Gran potencialidad de aplicación
- Muy populares en muchos campos
- Muy potentes en diversas aplicaciones
- Altas prestaciones a bajo coste

# Otros conceptos: Diversidad, Exploración y Explotación

diversidad genética

- **Asociada a las diferencias** entre los cromosomas de la población
- Falta de diversidad genética **cuando todos los individuos son parecidos**

convergencia al vecino más cercano

- **Alta presión selectiva**

falta de diversidad

# Otros conceptos: Diversidad, Exploración y Explotación

diversidad genética

## En la práctica es irreversible

son parecidos

convergencia al vecino más cercano

- Alta presión selectiva

falta de diversidad

# Otros conceptos: Diversidad, Exploración y Explotación

diversidad genética

## En la práctica es irreversible

son parecidos

convergen

- Alta presión selectiva

Mecanismos de reinicialización  
cuando tenemos una  
convergencia prematura

# Otros conceptos: Diversidad, Exploración y Explotación

diversidad genética

## En la práctica es irreversible

son parecidos

Incluir mecanismos para provocar diversidad en el proceso evolutivo

rgen

Mecanismos de reinicialización cuando tenemos una convergencia prematura

# Otros conceptos: Diversidad, Exploración y Explotación

diversidad genética

exploración

- Búsqueda en **zonas no exploradas**
- Una excesiva exploración nos conduciría a una búsqueda aleatoria, y difícil convergencia

explotación

- **Tratar de mejorar al individuo**
- Una excesiva explotación es similar a una búsqueda local, o converger a óptimos locales

# Ejercicio

SEUR tiene un problema de almacenamiento en sus instalaciones de Jaén. En concreto, SEUR maneja en un día concreto unos 1000 productos y se deben mantener siempre las restricciones de volumen ( $169750 \text{ m}^3$ ) y peso (545 kg)

El cliente necesita maximizar la siguiente función siendo  $i$  un elemento:

$$\textit{fitness} = \sum_{i=1}^n g_i p_i \text{ donde } x_i \in \{0,1\}, i = 1, \dots, n$$

Considera una tabla inicial con 10 productos asociando una ganancia (g), peso y (p) volumen para (v) cada uno ellos.

Diseña un algoritmo evolutivo para resolver este problema con una inicialización aleatoria, cruce en dos puntos y mutación aleatoria. Diseña el esquema generacional de este algoritmo con élite de dos individuos en una población de 100 individuos

# Ejercicio de la mochila

Ejercicio de programación  
lineal

Concepto más importante  
sería el orden de introducir  
elementos en la mochila

Imaginemos un peso  
máximo de 100

	valor	peso	valor/peso
1	8	10	0,8
2	18	20	1,75
3	66	30	2,2
4	12	40	0,3
5	50	50	1

# Ejercicio de la mochila

	valor	peso	valor/peso
1	8	10	0,8
2	18	20	1,75
3	66	30	2,2
4	12	40	0,3
5	35	50	0,7

Por peso menor entrarían el 1, 2, 3, 4  
Función maximización = 104

# Ejercicio de la mochila

	valor	peso	valor/peso
1	8	10	0,8
2	18	20	1,75
3	66	30	2,2
4	12	40	0,3
5	35	50	0,7

**Por peso menor entrarían el 1, 2, 3, 4**  
**Función maximización = 104**

**Por valor mayor entrarían el 3, 5, 2**  
**Función maximización = 119**

# Ejercicio de la mochila

	valor	peso	valor/peso
1	8	10	0,8
2	18	20	1,75
3	66	30	2,2
4	12	40	0,3
5	35	50	0,7

**Por peso menor entrarían el 1, 2, 3, 4**

**Función maximización = 104**

**Por valor mayor entrarían el 3, 5, 2**

**Función maximización = 119**

**Por valor/peso mayor**

**entrarían el 3, 2, 1**

**Función maximización = 92**



# Metaheurísticas

## Grado en Ingeniería Informática

### Universidad de Jaén

### Cristóbal J. Carmona

### Curso 2023/2024

Esta obra está protegida con licencia  
Creative Commons Atribución-NoComercial 4.0 Internacional

