

# Algoritmos Genéticos

Introducción a la Robótica Inteligente

---



Álvaro Gutiérrez  
20 de abril de 2022

[a.gutierrez@upm.es](mailto:a.gutierrez@upm.es)  
[www.robolabo.etsit.upm.es](http://www.robolabo.etsit.upm.es)

- ❶ Introducción
- ❷ Algoritmos Genéticos
- ❸ Conclusiones

---

## 1 Introducción

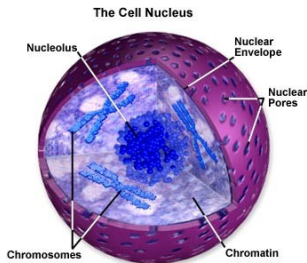
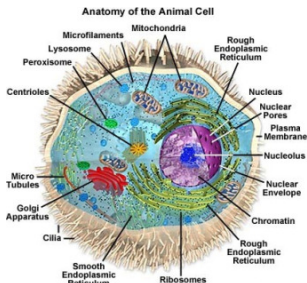
## 2 Algoritmos Genéticos

## 3 Conclusiones

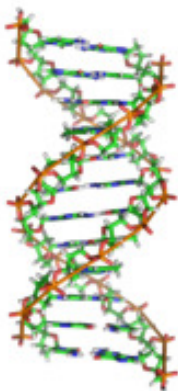
- Los **algoritmos genéticos** son algoritmos de búsqueda probabilística u **optimización** que transforman iterativamente un conjunto (llamado **población**) de objetos matemáticos, cada uno con un valor de “coste” (**fitness**) asociado, en una **nueva población** de descendientes usando principios **Darvinianos** de selección natural y usando **operaciones genéticas** naturales tales como “crossover” (reproducción sexual) y mutación.

- ▶ Primeras Ideas: John H. **Holland**
  - ▶ Adaptation in Natural and Artificial Systems
- ▶ Otros nombres: K. DeJong y D. **Goldberg**
- ▶ Típicamente usado en **optimización discreta**
- ▶ Características:
  - ▶ No son muy rápidos
  - ▶ Búsqueda en paralelo
- ▶ De un vistazo:
  - ▶ Una pila de soluciones
  - ▶ Combinar las soluciones existentes para producir nuevas soluciones
  - ▶ Mutar soluciones actuales para diversidad a largo plazo
  - ▶ Mantener las soluciones mejores y sacrificar las peores



- ▶ Todo animal está compuesto de células trabajando conjuntamente
- ▶ El centro de cada célula es el núcleo
- ▶ El núcleo contiene la información genética



- ▶ La **información** genética se almacena en los **cromosomas**
- ▶ Cada **cromosoma** está compuesto de **ADN**
- ▶ Los cromosomas en los humanos forman pares
- ▶ Los cromosomas están divididos en partes: **Genes**
- ▶ Cada **gen** puede adquirir diferentes valores: **alelos**
- ▶ Cada **gen** tiene una única posición (**locus**) en cada cromosoma

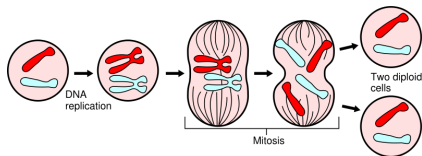
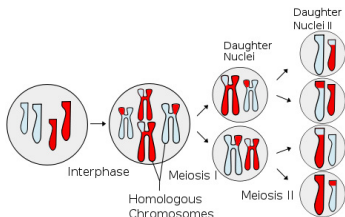


- ▶ El conjunto de todos los genes es un **genotipo**
- ▶ Cada genotipo desarrolla un **fenotipo**
- ▶ Los alelos pueden ser **dominantes** o **recesivos**
  - ▶ Los dominantes siempre se expresan en el fenotipo
  - ▶ Los recesivos pueden mantenerse durante generaciones sin “dar la cara”

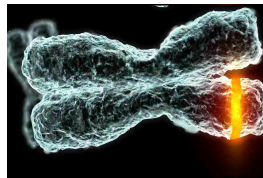
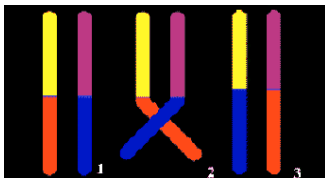
|   |   |   |   |   |  |   |
|---|---|---|---|---|--|---|
|  | + |  | = |  75% |  18.75% |  6.25% |
|  | + |  | = |  50% |  37.5%  |  12.5% |
|  | + |  | = |  50% |  0%     |  50%   |
|  | + |  | = |  <1% |  75%    |  25%   |
|  | + |  | = |  0%  |  50%    |  50%   |
|  | + |  | = |  0%  |  1%     |  99%   |



- **Meiosis:** Un tipo de reproducción celular en el que el número de cromosomas es reducido a la mitad separando cromosomas homólogos
- **Mitosis:** Un tipo de reproducción asexual en el que la célula se divide creando una réplica (copia exacta) con el mismo número de cromosomas



- ▶ Durante la reproducción ocurren combinaciones y errores
- ▶ Gracias a estas, la variedad existe
- ▶ Los más importantes:
  - ▶ Cross-over
  - ▶ Mutación



- ▶ Se **preservan** las variaciones **favorables** y se **rechazan** las variaciones **no favorables**
- ▶ Cada generación nacen nuevos individuos, por lo que existe una **lucha permanente**
- ▶ Los individuos con ventajas tienen una mayor posibilidad de supervivencia: **Supervivencia del más adecuado**
- ▶ Aspectos importantes:
  - ▶ **Adaptación** al entorno
  - ▶ **Aislamiento** de especies con las que no se puede reproducir

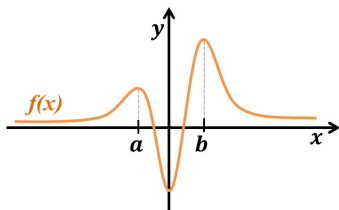
1 Introducción

**2 Algoritmos Genéticos**

3 Conclusiones

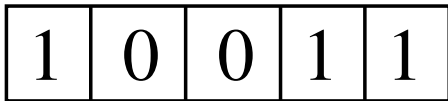
- ▶ Los AGs trabajan con una **codificación** del conjunto de parámetros, no con los parámetros mismos
- ▶ Los AGs buscan en un **conjunto de puntos**, no un único punto
- ▶ Los AGs utilizan una **función objetivo**, no derivadas, funcionales u otras funciones
- ▶ Los AGs utilizan **reglas** de transición **probabilística**, no determinísticas.

- ▶ Cada individuo busca la **mejor solución** en un conjunto
- ▶ Este espacio es el **espacio de búsqueda**
- ▶ Cada punto en el espacio de búsqueda es una posible solución
- ▶ Cada punto tiene un valor de “**fitness**”(encaje) asociado
- ▶ Los algoritmos genéticos buscan soluciones en **paralelo**
- ▶ Los problemas:
  - ▶ Óptimos locales
  - ▶ Condiciones iniciales



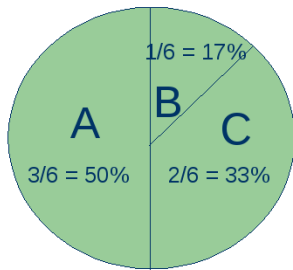
- ▶ Se comienza con una **población aleatoria** de **n individuos**
- ▶ Se **evalúa** cada individuo
- ▶ Se crea una **nueva generación**
  - ▶ **Selección**: Los mejores
  - ▶ **Recombinación**: Entre los mejores
  - ▶ **Mutación**: Aleatoria
- ▶ Se **evalúa** la nueva generación
- ▶ **Repetimos** para **m** generaciones

- ▶ Los **cromosomas** se codifican en **cadena de bits**
- ▶ Cada cromosoma representa un individuo
- ▶ Cada individuo es una solución, aunque no la mejor
- ▶ La codificación depende del problema a resolver

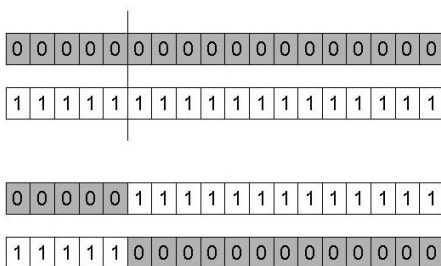




- ▶ Principal idea: Los **mejores** tienen **más posibilidades** de ser seleccionados
- ▶ Típicamente la **ruleta**
  - ▶ Asigna a **cada individuo una parte** de la ruleta
  - ▶ **Girar** la ruleta **n** veces para crear una población de n individuos



- ▶ Se seleccionan 2 individuos
- ▶ Se realiza un cruce con probabilidad  $P_c$
- ▶  $P_c$  típicamente en el rango (0.6, 0.9)
- ▶ Se selecciona un punto de cruce aleatorio



- ▶ Alterar cada gen con probabilidad  $P_m$
- ▶  $P_m$  típicamente en el rango  $(\frac{1}{Long.Poblacion}, \frac{1}{Long.Cromosoma})$

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

- ▶ Un ejemplo sencillo:  $\max(x^2)$  donde  $x \in \{0, 1, \dots, 31\}$
- ▶ Algoritmo genético
  - ▶ Codificación en 5 bits, e.g.  $01101 \leftrightarrow 13$
  - ▶ Población de 4 individuos
  - ▶ Inicio aleatorio
  - ▶ Selección por ruleta
  - ▶ Crossover
  - ▶ Mutación

# Un Primer Ejemplo - Selección



| String no. | Initial population | $x$ Value | Fitness $f(x) = x^2$ | $Prob_i$ | Expected count | Actual count |
|------------|--------------------|-----------|----------------------|----------|----------------|--------------|
| 1          | 0 1 1 0 1          | 13        | 169                  | 0.14     | 0.58           | 1            |
| 2          | 1 1 0 0 0          | 24        | 576                  | 0.49     | 1.97           | 2            |
| 3          | 0 1 0 0 0          | 8         | 64                   | 0.06     | 0.22           | 0            |
| 4          | 1 0 0 1 1          | 19        | 361                  | 0.31     | 1.23           | 1            |
| Sum        |                    |           | 1170                 | 1.00     | 4.00           | 4            |
| Average    |                    |           | 293                  | 0.25     | 1.00           | 1            |
| Max        |                    |           | 576                  | 0.49     | 1.97           | 2            |

# Un Primer Ejemplo - Crossover



| String no. | Mating pool | Crossover point | Offspring after xover | $x$ Value | Fitness $f(x) = x^2$ |
|------------|-------------|-----------------|-----------------------|-----------|----------------------|
| 1          | 0 1 1 0   1 | 4               | 0 1 1 0 0             | 12        | 144                  |
| 2          | 1 1 0 0   0 | 4               | 1 1 0 0 1             | 25        | 625                  |
| 2          | 1 1   0 0 0 | 2               | 1 1 0 1 1             | 27        | 729                  |
| 4          | 1 0   0 1 1 | 2               | 1 0 0 0 0             | 16        | 256                  |
| Sum        |             |                 |                       |           | 1754                 |
| Average    |             |                 |                       |           | 439                  |
| Max        |             |                 |                       |           | 729                  |

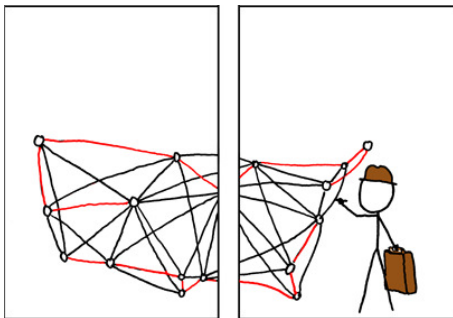
# Un Primer Ejemplo - Mutación



| String no. | Offspring after xover | Offspring after mutation | $x$ Value | Fitness $f(x) = x^2$ |
|------------|-----------------------|--------------------------|-----------|----------------------|
| 1          | 0 1 1 0 0             | 1 1 1 0 0                | 26        | 676                  |
| 2          | 1 1 0 0 1             | 1 1 0 0 1                | 25        | 625                  |
| 2          | 1 1 0 1 1             | 1 1 0 1 1                | 27        | 729                  |
| 4          | 1 0 0 0 0             | 1 0 1 0 0                | 18        | 324                  |
| Sum        |                       |                          |           | 2354                 |
| Average    |                       |                          |           | 588.5                |
| Max        |                       |                          |           | 729                  |

## Otro ejemplo sencillo - TSP

- ▶ El problema del **vendedor viajero** (Travelling Salesman Problem)
- ▶ Dado un conjunto de ciudades encontrar un recorrido de tal manera que:
  - ▶ Cada ciudad sólo se visite una vez
  - ▶ La distancia recorrida se minimice



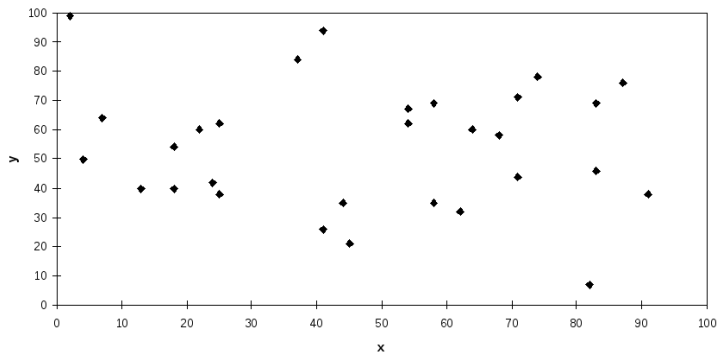


- Representación en una lista ordenada

|            |             |               |             |
|------------|-------------|---------------|-------------|
| 1) Londres | 3) Madrid   | 5) Pekín      | 7) Tokio    |
| 2) Venecia | 4) Singapur | 6) Nueva York | 8) El Cairo |

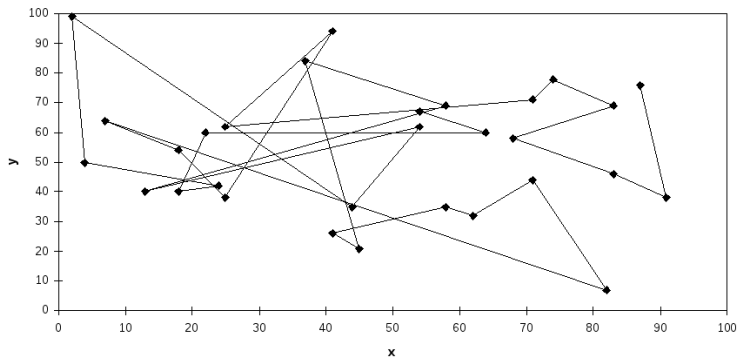
- Individuo1: ( 3 5 7 2 1 6 4 8 )
- Individuo2: ( 2 5 7 6 8 1 3 4 )
- ...

## ► Generación 0



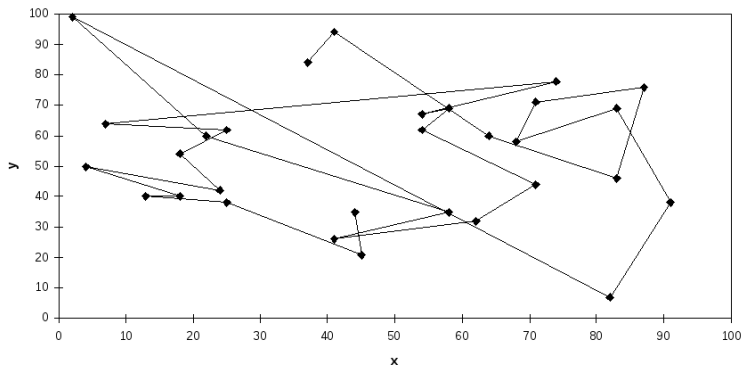
## ► Generación 1

TSP30 (Performance = 941)



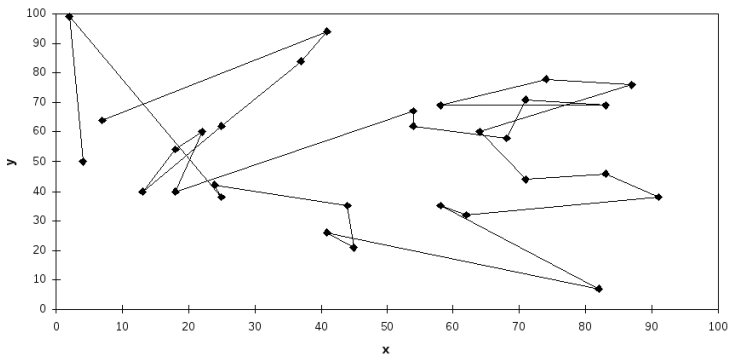
## ► Generación 30

TSP30 (Performance = 800)



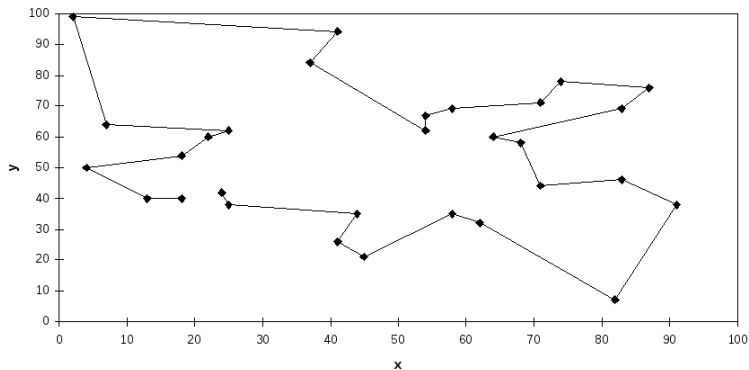
## ► Generación 43

TSP30 (Performance = 652)



## ► Generación 100

TSP30 Solution (Performance = 420)



1 Introducción

2 Algoritmos Genéticos

**3 Conclusiones**

- ▶ Problemas de los AGs:
  - ▶ Hay que elegir demasiadas cosas:
    - ▶ representación
    - ▶ tamaño de la población, prob. de crossover, prob. de mutación,...
    - ▶ operadores de selección, crossover, mutación,...
  - ▶ Escalabilidad
  - ▶ La solución sólo es tan buena como la función de “fitness”
    - ▶ Normalmente la parte más difícil



- ▶ Beneficio de los AGs:
  - ▶ Sencillo de entender
  - ▶ Modular, separado de la aplicación
  - ▶ Permite optimización multi-objetivo
  - ▶ Bueno en entornos con ruido
  - ▶ Siempre hay una solución
  - ▶ Distribuido, paralelo,...

**GRACIAS!!**

**GRACIAS!!**