

ALGORITMOS EVOLUTIVOS

Tecnología Digital V: Diseño de Algoritmos
Universidad Torcuato Di Tella

- Un **algoritmo genético** busca una solución a un problema haciendo evolucionar una población inicial de candidatos a solución.

- Un **algoritmo genético** busca una solución a un problema haciendo evolucionar una población inicial de candidatos a solución.
- Cada iteración corresponde a una generación, en la que sobreviven las mejores soluciones.

- Un **algoritmo genético** busca una solución a un problema haciendo evolucionar una población inicial de candidatos a solución.
- Cada iteración corresponde a una generación, en la que sobreviven las mejores soluciones.
- Las soluciones se representan en forma binaria por medio de **cadenas de bits**:

0001010110101001010

- Un **algoritmo genético** busca una solución a un problema haciendo evolucionar una población inicial de candidatos a solución.
- Cada iteración corresponde a una generación, en la que sobreviven las mejores soluciones.
- Las soluciones se representan en forma binaria por medio de **cadenas de bits**:

0001010110101001010

- Se tienen operadores de **mutación** y **recombinación** para generar nuevas soluciones a partir de las soluciones existentes.

- Un **algoritmo evolutivo** es un algoritmo genético en el que las soluciones no se representan en forma binaria.

- Un **algoritmo evolutivo** es un algoritmo genético en el que las soluciones no se representan en forma binaria.
 1. En lugar de utilizar una representación binaria, se utiliza una representación lo más adecuada posible para el problema en cuestión.

- Un **algoritmo evolutivo** es un algoritmo genético en el que las soluciones no se representan en forma binaria.
 1. En lugar de utilizar una representación binaria, se utiliza una representación lo más adecuada posible para el problema en cuestión.
 2. Esto permite más **flexibilidad** para trabajar con las soluciones.

- Un **algoritmo evolutivo** es un algoritmo genético en el que las soluciones no se representan en forma binaria.
 1. En lugar de utilizar una representación binaria, se utiliza una representación lo más adecuada posible para el problema en cuestión.
 2. Esto permite más **flexibilidad** para trabajar con las soluciones.
 3. Como contrapartida, se deben redefinir para cada caso los operadores de construcción aleatoria, mutación y recombinación.

- **Ejemplo:** El problema del viajante de comercio. Recordemos ...

- **Ejemplo:** El problema del **viajante de comercio**. Recordemos ...
- Dado un conjunto de ciudades, el problema del viajante de comercio consiste en determinar un recorrido que pase por todas las ciudades y minimice la distancia total.

- **Ejemplo:** El problema del **viajante de comercio**. Recordemos ...
- Dado un conjunto de ciudades, el problema del viajante de comercio consiste en determinar un recorrido que pase por todas las ciudades y minimice la distancia total.
 1. **Entrada:** La cantidad n de ciudades y una matriz simétrica $A \in \mathbb{R}^{n \times n}$ de distancias entre las ciudades.

- **Ejemplo:** El problema del **viajante de comercio**. Recordemos ...
- Dado un conjunto de ciudades, el problema del viajante de comercio consiste en determinar un recorrido que pase por todas las ciudades y minimice la distancia total.
 1. **Entrada:** La cantidad n de ciudades y una matriz simétrica $A \in \mathbb{R}^{n \times n}$ de distancias entre las ciudades.
 2. **Salida:** En qué orden se deben recorrer las ciudades para minimizar la distancia total.

- Una solución está dada por una permutación de las ciudades. Se puede utilizar una representación binaria, con n genes:

$$\underbrace{0110101101}_{\text{ciudad 1}} \underbrace{1011011010}_{\text{ciudad 2}} \underbrace{0001110100}_{\text{ciudad 3}} \dots$$

- Una solución está dada por una permutación de las ciudades. Se puede utilizar una representación binaria, con n genes:

$$\underbrace{0110101101}_{\text{ciudad 1}} \underbrace{1011011010}_{\text{ciudad 2}} \underbrace{0001110100}_{\text{ciudad 3}} \dots$$

- >Qué problema tiene esta representación?

- Una solución está dada por una permutación de las ciudades. Se puede utilizar una representación binaria, con n genes:

$$\underbrace{0110101101}_{\text{ciudad 1}} \underbrace{1011011010}_{\text{ciudad 2}} \underbrace{0001110100}_{\text{ciudad 3}} \dots$$

- >Qué problema tiene esta representación?
- Las mutaciones y recombinaciones estándar pueden no generar una solución válida!

- Representamos ahora una solución por medio de un arreglo de enteros que contenga los números (permutados) de 1 a n :

4 3 1 6 2 5 7

- Representamos ahora una solución por medio de un arreglo de enteros que contenga los números (permutados) de 1 a n :

4 3 1 6 2 5 7

- La permutación representa las ciudades en orden del recorrido.

- Representamos ahora una solución por medio de un **arreglo de enteros** que contenga los números (permutados) de 1 a n :

4 3 1 6 2 5 7

- La permutación representa las ciudades en orden del recorrido.
 1. Es sencillo **generar aleatoriamente** nuevas soluciones (>cómo lo hacemos?).

- Representamos ahora una solución por medio de un **arreglo de enteros** que contenga los números (permutados) de 1 a n :

4 3 1 6 2 5 7

- La permutación representa las ciudades en orden del recorrido.
 1. Es sencillo **generar aleatoriamente** nuevas soluciones (>cómo lo hacemos?).
 2. Evaluar la **función de fitness** sigue siendo $O(n)$ con esta nueva representación.

- Para generar una mutación de una solución, podemos seleccionar dos ciudades aleatoriamente e **intercambiarlas**:

4 3 **1** 6 2 **5** 7 \leadsto 4 3 **5** 6 2 **1** 7

- Para generar una mutación de una solución, podemos seleccionar dos ciudades aleatoriamente e **intercambiarlas**:

4 3 **1** 6 2 **5** 7 \leadsto 4 3 **5** 6 2 **1** 7

- Para recombinar dos soluciones, podemos **componer las permutaciones**! Esto corresponde a permutar la primera solución de acuerdo con la especificación de la segunda solución:

4 3 1 6 2 5 7
5 4 2 3 7 6 1 \leadsto **2 6 3 1 7 5 4**

- En un algoritmo evolutivo, para cada problema se debe pensar una representación adecuada para las soluciones, y los mecanismos de mutación y recombinación.

- En un algoritmo evolutivo, para cada problema se debe pensar una representación adecuada para las soluciones, y los mecanismos de mutación y recombinación.
- Habitualmente, es posible definir más de un mecanismo de mutación o recombinación, y para determinar cuál es el mejor hay que hacer experimentos sobre **bancos de prueba**.

- En un algoritmo evolutivo, para cada problema se debe pensar una representación adecuada para las soluciones, y los mecanismos de mutación y recombinación.
- Habitualmente, es posible definir más de un mecanismo de mutación o recombinación, y para determinar cuál es el mejor hay que hacer experimentos sobre **bancos de prueba**.
- También es posible definir más de una representación para las soluciones, y nuevamente se debe definir cuál tiene mejor performance por medio de experimentos.