## Optimización de funciones de De Jong

De Jong es un investigador que diseñó un conjunto de cinco problemas para evaluar algoritmos de optimización de funciones, que se conocen como las funciones De Jong. Estas funciones tienen diferentes características, como ser continuas o discontinuas, convexas o no convexas, unimodales o multimodales, cuadráticas o no cuadráticas, y tener baja o alta dimensionalidad. Además, algunas de estas funciones son determinísticas, mientras que otras son estocásticas.

El propósito de utilizar estas funciones es evaluar el desempeño de los algoritmos de optimización, ya que permiten conocer su capacidad para encontrar el mínimo global de una función en diferentes situaciones. Cada función De Jong tiene una solución óptima conocida, lo que permite comparar los resultados obtenidos por diferentes algoritmos de optimización. Estas funciones son ampliamente utilizadas como benchmarks para evaluar y comparar diferentes técnicas de optimización, incluyendo la optimización por algoritmos genéticos.

## Problema del viajante

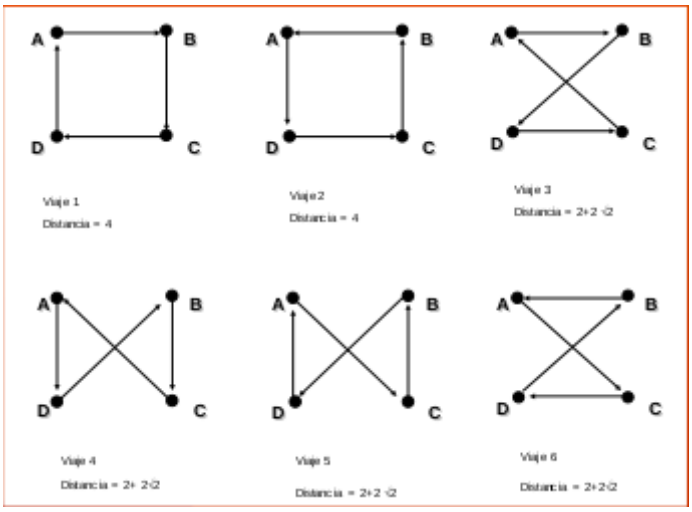
El problema del viajante (TSP, por sus siglas en inglés) es un problema de optimización combinatoria en el que se busca encontrar el recorrido más corto que pasa por todas las ciudades de una lista y regresa a la ciudad de origen. Este problema es NP-completo, lo que significa que no hay una solución algorítmica eficiente para resolverlo en tiempo polinómico.

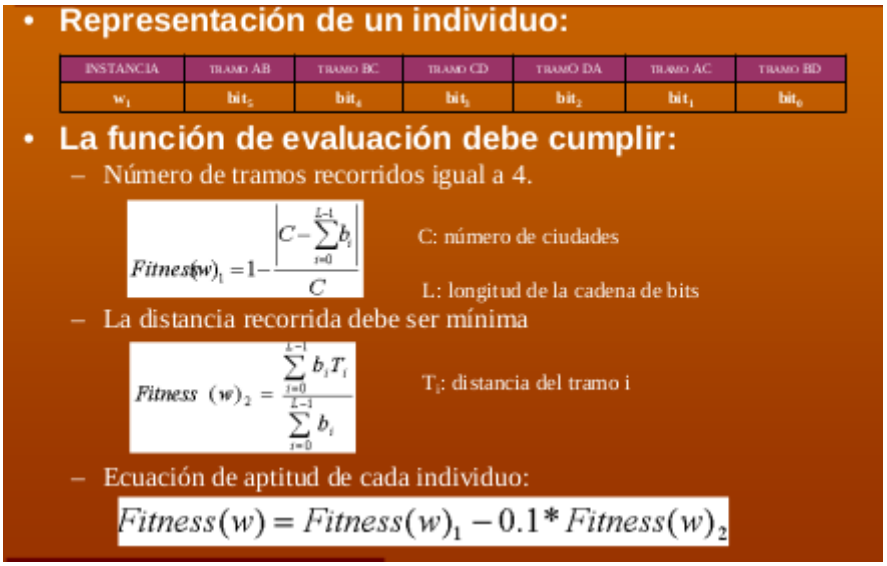
Una forma de resolver el problema del viajante es utilizando algoritmos genéticos (AGs). Los pasos básicos para resolverlo con AGs son los siguientes:

1. Crear una población inicial: se genera un conjunto de recorridos aleatorios para la población inicial.
2. Seleccionar los dos mejores individuos de la población y combinarlos: se eligen los dos recorridos con menor distancia y se realiza un operador de cruce para generar nuevos individuos.
3. Modificar los operadores de cruce y mutación: para el problema del viajante se utilizan operadores específicos. El cruce se realiza mediante la técnica de permutaciones de ciudades y la mutación cambia el orden de algunas ciudades del recorrido.
4. Repetir los pasos 2 y 3 hasta que se alcance un criterio de parada: se seleccionan los dos mejores individuos de la población, se aplican los operadores de cruce y mutación, y se repite el proceso hasta alcanzar un criterio de parada.

A veces, el algoritmo puede converger a un punto en el que todos los individuos son iguales. Para evitar esto, se puede partir de una población inicial grande y utilizar el método de mutación para introducir diversidad en la población.

Por ejemplo, para 4 ciudades:




## Problema de satisfacción: forma normal conjuntiva

Las FNC (Formas Normales Conjuntivas) de la lógica son una secuencia de cláusulas unidas por la relación AND. Cada cláusula es una disyunción de literales, es decir, variables booleanas o su negación. El problema de satisfacción de FNC consiste en encontrar valores de verdad de los literales que hagan que la secuencia completa se evalúe a verdadero (true).

Para representar las posibles soluciones, se utiliza una cadena de bits, donde cada bit indica el valor true o false de cada literal. Por ejemplo, si tenemos la secuencia de literales (a OR NOT b OR c), podemos representarla con la cadena de bits 101.

En la computación evolutiva, se busca utilizar operadores genéticos para generar descendientes que evalúen a cierto la expresión. Para ello, se utiliza una población de cadenas

de bits (posibles soluciones) que se va evolucionando mediante operadores como la selección, cruce y mutación. El objetivo es encontrar la solución óptima, es decir, la cadena de bits que satisfaga la FNC de forma más eficiente posible.

Por ejemplo, si tenemos una expresión FNC con 6 cláusulas y se representa como una cadena de 6 bits, donde cada bit indica el valor de verdad de cada literal de la cláusula, entonces una posible solución con fitness 3 podría ser 010011. Esto indica que tres cláusulas de la expresión satisfacen sus literales. El objetivo sería encontrar una cadena de bits que tenga un valor máximo de fitness, que en este caso sería 6, lo que significaría que todas las cláusulas se satisfacen. Por ejemplo: