Diploma Programme
Programme du diplôme
Programa del Diploma

# Informática
# Estudio de caso: Algoritmos genéticos

Para usar en noviembre de 2021, mayo de 2022 y noviembre de 2022

---

**Instrucciones para los alumnos**

- Para la prueba 3 de Nivel Superior se requiere el cuadernillo del estudio de caso.

## Introducción

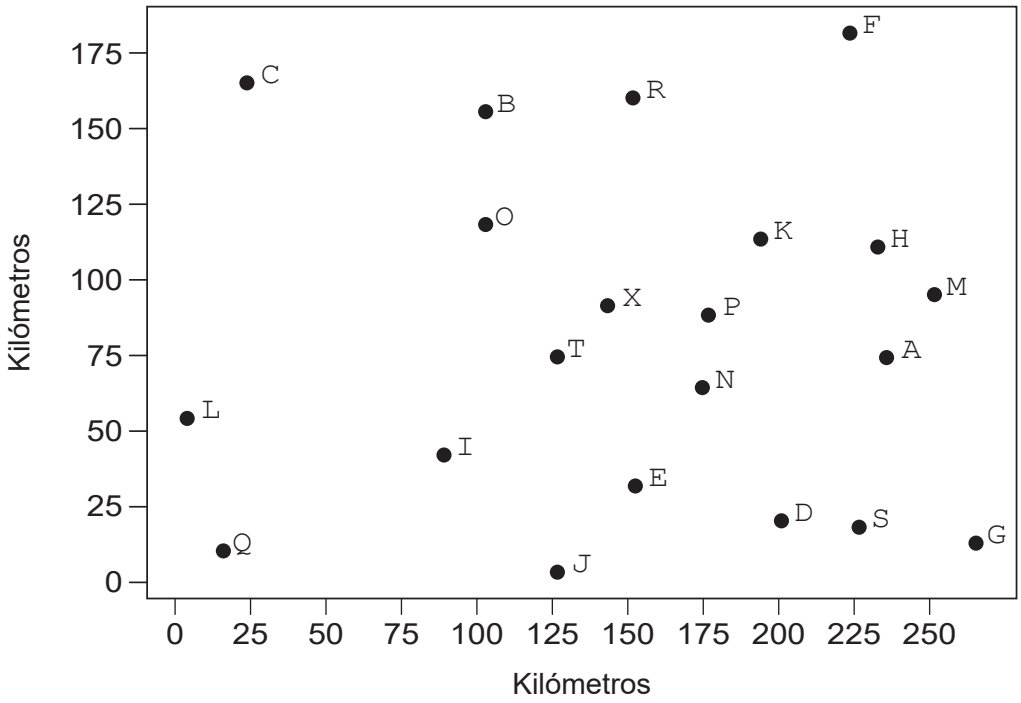
Son las siete de un sábado por la noche y Lotte está empacando sus cosas antes de partir a un recorrido en motocicleta por Vlakland, que comenzará el lunes por la mañana. Ha identificado 20 ciudades que quiere visitar antes de volver. Su amiga, Fenna, que está el segundo año del curso de Informática del Programa del Diploma, le pregunta qué ruta va a seguir. “La verdad es que esperaba que pudieras ayudarme con eso”, responde Lotte. “He encontrado una tabla de las distancias entre las ciudades que quiero visitar. Querría saber si podrías escribir un breve programa en la computadora para probar todas las rutas posibles” (ver la **Figura 1**).

**Figura 1: Distancias en kilómetros entre las ciudades que Lotte quiere visitar**

| | X | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **X** | 0 | 94 | 76 | 141 | 91 | 60 | 120 | 145 | 91 | 74 | 90 | 55 | 145 | 108 | 41 | 49 | 33 | 151 | 69 | 111 | 24 |
| **A** | 94 | 0 | 156 | 231 | 64 | 93 | 108 | 68 | 37 | 150 | 130 | 57 | 233 | 26 | 62 | 140 | 61 | 229 | 120 | 57 | 109 |
| **B** | 76 | 156 | 0 | 80 | 167 | 133 | 124 | 216 | 137 | 114 | 154 | 100 | 141 | 161 | 116 | 37 | 100 | 169 | 49 | 185 | 84 |
| **C** | 141 | 231 | 80 | 0 | 229 | 185 | 201 | 286 | 216 | 139 | 192 | 178 | 113 | 239 | 182 | 92 | 171 | 155 | 128 | 251 | 137 |
| **D** | 91 | 64 | 167 | 229 | 0 | 49 | 163 | 65 | 96 | 114 | 76 | 93 | 200 | 91 | 51 | 139 | 72 | 185 | 148 | 26 | 92 |
| **E** | 60 | 93 | 133 | 185 | 49 | 0 | 165 | 115 | 112 | 65 | 39 | 91 | 151 | 117 | 39 | 99 | 61 | 139 | 128 | 75 | 49 |
| **F** | 120 | 108 | 124 | 201 | 163 | 165 | 0 | 173 | 71 | 194 | 203 | 74 | 254 | 90 | 127 | 136 | 104 | 269 | 75 | 163 | 144 |
| **G** | 145 | 68 | 216 | 286 | 65 | 115 | 173 | 0 | 103 | 179 | 139 | 123 | 265 | 83 | 104 | 194 | 116 | 250 | 186 | 39 | 152 |
| **H** | 91 | 37 | 137 | 216 | 96 | 112 | 71 | 103 | 0 | 160 | 151 | 39 | 236 | 25 | 75 | 130 | 61 | 239 | 95 | 93 | 112 |
| **I** | 74 | 150 | 114 | 139 | 114 | 65 | 194 | 179 | 160 | 0 | 54 | 127 | 86 | 171 | 89 | 77 | 99 | 80 | 134 | 140 | 50 |
| **J** | 90 | 130 | 154 | 192 | 76 | 39 | 203 | 139 | 151 | 54 | 0 | 129 | 133 | 155 | 78 | 117 | 99 | 111 | 159 | 101 | 71 |
| **K** | 55 | 57 | 100 | 178 | 93 | 91 | 74 | 123 | 39 | 127 | 129 | 0 | 199 | 61 | 53 | 91 | 30 | 206 | 63 | 101 | 78 |
| **L** | 145 | 233 | 141 | 113 | 200 | 151 | 254 | 265 | 236 | 86 | 133 | 199 | 0 | 251 | 171 | 118 | 176 | 46 | 182 | 226 | 125 |
| **M** | 108 | 26 | 161 | 239 | 91 | 117 | 90 | 83 | 25 | 171 | 155 | 61 | 251 | 0 | 83 | 151 | 75 | 251 | 119 | 81 | 127 |
| **N** | 41 | 62 | 116 | 182 | 51 | 39 | 127 | 104 | 75 | 89 | 78 | 53 | 171 | 83 | 0 | 90 | 24 | 168 | 99 | 69 | 49 |
| **O** | 49 | 140 | 37 | 92 | 139 | 99 | 136 | 194 | 130 | 77 | 117 | 91 | 118 | 151 | 90 | 0 | 80 | 139 | 65 | 159 | 50 |
| **P** | 33 | 61 | 100 | 171 | 72 | 61 | 104 | 116 | 61 | 99 | 99 | 30 | 176 | 75 | 24 | 80 | 0 | 179 | 76 | 86 | 52 |
| **Q** | 151 | 229 | 169 | 155 | 185 | 139 | 269 | 250 | 239 | 80 | 111 | 206 | 46 | 251 | 168 | 139 | 179 | 0 | 202 | 211 | 128 |
| **R** | 69 | 120 | 49 | 128 | 148 | 128 | 75 | 186 | 95 | 134 | 159 | 63 | 182 | 119 | 99 | 65 | 76 | 202 | 0 | 161 | 90 |
| **S** | 111 | 57 | 185 | 251 | 26 | 75 | 163 | 39 | 93 | 140 | 101 | 101 | 226 | 81 | 69 | 159 | 86 | 211 | 161 | 0 | 115 |
| **T** | 24 | 109 | 84 | 137 | 92 | 49 | 144 | 152 | 112 | 50 | 71 | 78 | 125 | 127 | 49 | 50 | 52 | 128 | 90 | 115 | 0 |

**Nota:** X es la casa de Lotte

**Figura 2: Imagen generada por computadora que utiliza los datos de la Figura 1 para mostrar la ubicación de las ciudades que Lotte desea visitar**




Fenna frunce el ceño. Es que ha reconocido la propuesta de Lotte como un ejemplo de un problema de *optimización combinatoria,* al que se conoce como el *problema del vendedor ambulante*. En este, el número de soluciones posibles crece de forma extremadamente rápida con el tamaño de la entrada, de modo que incluso problemas bastante pequeños, como visitar 20 ciudades diferentes por la ruta más corta posible, se vuelven *computacionalmente intratables*.

"Digamos que tu punto de partida es X y las ciudades que deseas visitar están etiquetadas como A, B, C, etc.", explica Fenna. "Si solo visitas una ciudad, solo hay una opción, que es de X hasta A y viceversa. Podemos escribir esto como XAX. Si tienes dos ciudades, entonces la secuencia es XABX o XBAX. Si hay tres ciudades, entonces tienes seis opciones:

XABCX XACBX XBACX XBCAX XCABX XCBAX".

"Sin embargo, XABX y XBAX son la misma ruta, ¡pero en direcciones diferentes!", observa Lotte.

"Así es", responde Fenna. "El número de permutaciones siempre se reduce a la mitad, porque cada ruta ocurre dos veces, una en cada una de las dos direcciones. El problema es el siguiente: cada vez que agregamos una nueva ciudad, se la puede insertar en cualquier punto de todas las rutas posibles actuales. Agregar la enésima ciudad multiplica el número de soluciones existentes por N".

"¡Pero no tenemos por qué hacerlo a mano!" dice Lotte, entre carcajadas. "¡Tenemos una computadora!"

"Bueno, sí", acuerda Fenna, "pero con 20 ubicaciones tenemos $20!$ permutaciones". Abre la aplicación de la calculadora en su teléfono. "Eso es un total de 1 216 451 004 088 320 000 posibilidades". Fenna hace cálculos en su teléfono y finalmente levanta la vista: "Todavía llevaría más de 30 000 años probar todas las permutaciones".

**Véase al dorso**

### El problema del vendedor ambulante

Implica empezar en una ciudad y visitar todas las demás ciudades antes de regresar al punto de partida. Todas las ciudades están conectadas entre sí por una ruta directa, todas deben visitarse una vez, y ninguna ciudad puede visitarse más de una vez. Cualquier secuencia de ciudades que obedezca estas reglas es un *recorrido* válido. El objetivo del problema en su versión más estricta es encontrar el recorrido más corto posible.

Hay una variedad de enfoques al problema del vendedor ambulante, muchos de los cuales requieren considerables conocimientos de matemáticas, pero actualmente no se sabe si existe un algoritmo que encuentre la solución óptima en un tiempo razonable. Sin embargo, hay recursos *heurísticos* que han tenido cierto éxito en encontrar buenas soluciones de una forma lo suficientemente rápida como para poder usarse en la práctica. Una de estas es el tema de este estudio de caso.

## Algoritmos genéticos

Los algoritmos genéticos imitan el proceso de selección natural en un intento de desarrollar soluciones a problemas que de otra manera serían intratables computacionalmente.

Los detalles de implementación varían considerablemente, pero un algoritmo genético estándar incluye los siguientes pasos:

```
Inicializar
While true
    Evaluar
    If (condición de terminación) detener
    Seleccionar
    Cruce
    Mutar
Output mejor resultado
```

El resto de esta sección examina algunas opciones de implementación al aplicar algoritmos genéticos al problema del vendedor ambulante.

### Inicialización

Implica generar una *población* aleatoria de recorridos individuales, cada uno de los cuales es una posible solución al problema. En el problema del vendedor ambulante, con un punto de partida de `X`, un posible recorrido es:

| F | J | G | I | L | C | M | E | S | Q | P | H | T | B | K | N | R | D | O | A |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

### Evaluación

El algoritmo determina si se ha cumplido la *condición de terminación* y, de ser así, genera la mejor solución encontrada hasta el momento y finaliza. Caso contrario, determina la *aptitud* de cada recorrido individual. En el problema de Lotte, esto implica calcular la distancia total que ella viajaría usando ese recorrido.

Se utiliza una *función de aptitud* para asignar un valor de aptitud a cada recorrido. Se clasifican los recorridos individuales según su aptitud. Se asigna el valor de aptitud más alto al recorrido más corto.

**Selección**

Se toma una muestra de recorridos del conjunto y se coloca en el *grupo de apareamiento*. Se puede hacer de varias maneras diferentes, pero, en general, los recorridos aptos tienen una mayor probabilidad de ser seleccionados para ingresar al grupo de apareamiento. Las cuatro *estrategias de selección* comunes son:

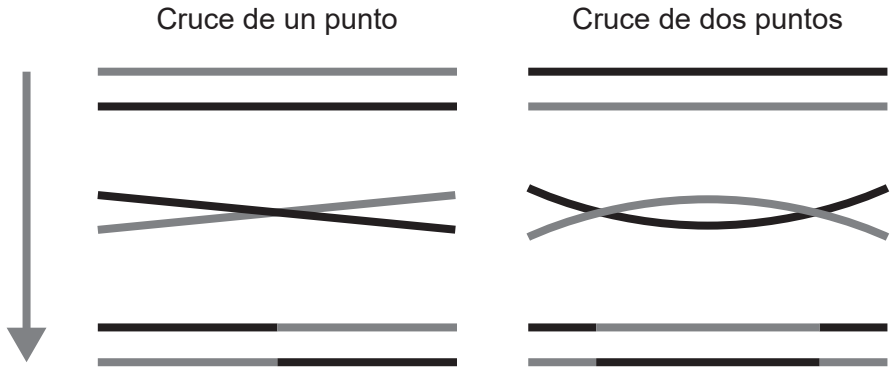
- *selección de rueda de ruleta*
- *muestreo universal estocástico*
- *selección de torneo*
- *selección de truncamiento*.

Una consideración de diseño adicional en la selección es decidir si asegurar la certeza de que a la próxima generación se lleve la mejor solución o soluciones. Esto se conoce como *elitismo*.

## Cruce

En biología, el *cruce* es el término dado al mecanismo por el cual se crean los cromosomas de un nuevo recorrido individual a partir de una combinación de los cromosomas de sus padres.

**Figura 3: Tipos de cruce**




En el problema del vendedor ambulante, el cruce de uno o de dos puntos (**Figura 3**) presenta un problema porque las ciudades pueden repetirse u omitirse en la *descendencia*, lo que lleva a un recorrido no válido que no visita cada ciudad según se requiere.

Considere el siguiente cruce (**Figura 4**) entre dos recorridos válidos de un problema de vendedor ambulante de diez ciudades, en el que se combinan los padres, `P1` y `P2`, usando un cruce de un punto para crear un nuevo recorrido individual, `F1`.

**Figura 4: Un ejemplo de un cruce**

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **P1** | **B** | **F** | **C** | **A** | **D** | **H** | **G** | **I** | **E** | **J** |
| **P2** | G | J | C | D | I | A | E | B | F | H |
| F1 | **B** | **F** | **C** | **A** | **D** | A | E | B | F | H |

**Véase al dorso**

La descendencia resultante, F1, no es un recorrido válido porque repite las ciudades A, B y F y omite las ciudades G, I y J. El mismo problema ocurre con un simple cruce de dos puntos. Existen varios mecanismos de cruce diferentes, y cada uno trata de preservar las características de los padres en la medida en que sea posible.

Después de decidir qué recorridos individuales conformarán el grupo de apareamiento, es necesario determinar cómo deben combinarse para producir descendencia. Se presentan tres métodos:

- Cruce parcialmente mapeado (PMX)
- Cruce en orden (OX)
- Cruce de ciclo (CX)

**Cruce parcialmente mapeado (PMX)**

Elija una subsecuencia aleatoria de P1 y cópiela a F1:

```
P1: J B F C A D H G I E
P2: F A G D H C E B J I

F1: * * F C A D H * * *
```

Configure mapeos por elementos entre P1 y P2 para ciudades en P1 que aún no están en F1. Si la ciudad correspondiente C de P2 ya está en F1, entonces reanude el mapeo desde la ubicación de C en P1, y repita hasta que se encuentre una ciudad que no esté en F1:

J ↔ G  B ↔ E  G ↔ B  I ↔ J  E ↔ I

Agregue las ciudades restantes de P1 a F1 y cámbielas de acuerdo con los mapeos:

```
P1: J B F C A D H G I E
P2: F A G D H C E B J I

F1: * * F C A D H * * *
    ↓ ↓           ↓ ↓ ↓
F1: J B F C A D H G I E
    ↓ ↓           ↓ ↓ ↓
F1: G E F C A D H B J I
```

**Cruce en orden (OX)**

Elija una subsecuencia de uno de los padres y conserve el orden relativo de las ciudades restantes de la otra.

Tome una subsecuencia aleatoria S de P1 y cópiela a F1. Al comenzar con el elemento vacío justo después de S en F1, copie todas las ciudades que aún no están en F1 desde P2 en el orden en que aparecen en P2.

```
P1: J B F C A D H G I E
P2: F A G D H C E B J I

F1: * B F C A D * * * *
    H B F C A D E J I G
```

**Cruce de ciclo (CX)**

En `F1`, cada ciudad mantiene la posición que tenía en al menos uno de sus padres.

**`P1:`** `J B F C A D H G I E`
`P2: F A G D H C E B J I`

Elija la primera ciudad de `P1` y cópiela a `F1`. Verifique la ciudad correspondiente en `P2` (en este caso, es la ciudad `F`) y cópiela a `F1`, en la misma posición en que aparece en `P1`. Repita.

`F1:` **`J`** `B F * A * H G I E`

Cuando encuentre una ciudad que ya está en `F1`, el ciclo se completa. Ahora complete las ciudades al copiar desde `P2`.

`F1: J B F` **`D`** `A` **`C`** `H G I E`

## Mutación

En biología, la *mutación se* refiere a errores accidentales en la copia de información genética de una generación a la siguiente. En un algoritmo genético, las mutaciones se introducen deliberadamente en cada nueva descendencia de acuerdo con la *tasa de mutación*.

## Discusión

Se cree que las ventajas de los algoritmos genéticos son su capacidad para muestrear simultáneamente amplios *panoramas de aptitud* al escaparse de los *extremos locales* donde los enfoques más tradicionales como el algoritmo de *escalada simple* (también conocido como “de ascenso de colinas”, o *hill climbing*) podrían quedar entrampados. Las implementaciones exitosas de algoritmos genéticos logran un equilibrio natural entre *la exploración* y la *explotación*, y técnicas como el algoritmo de *recocido simulado* (también llamado “templado simulado” o *simulated annealing*) pueden afinar ese equilibrio a medida que el algoritmo avanza hacia la *convergencia*. La investigación más reciente se ha centrado en premiar la *novedad* específicamente como un medio para alentar a los algoritmos a explorar regiones remotas del *espacio del problema*. Otras opciones de diseño, tales como los parámetros iniciales, la *estrategia de selección* y el *operador de cruce*, influyen en el rendimiento del algoritmo, aunque generalmente no es posible predecir sus efectos, por lo que a menudo se adopta un enfoque de prueba y error.

## Desafíos que se plantean

Hay una serie de desafíos asociados con los algoritmos genéticos. Estos incluyen:

- Comprender el papel de la convergencia en los algoritmos genéticos y los factores que la afectan.
- Evaluar el uso y la implementación de estrategias de selección de ruleta, de torneo y de truncamiento utilizadas en algoritmos genéticos.
- Discutir las diferentes soluciones para abordar el fracaso de las estrategias de cruce simple en el problema del vendedor ambulante. En particular:
  - por qué son necesarios
  - cómo se aplican
  - cómo preservan los rasgos parentales
  - qué otros métodos posibles están disponibles
- Comprender las ventajas y desventajas de los algoritmos genéticos con respecto a otros enfoques del problema del vendedor ambulante y los de optimización combinatoria en general.

**No se requiere que los candidatos conozcan los detalles de implementación de otros enfoques.**

**Véase al dorso**

## Terminología adicional a la de la guía

Aptitud/función de aptitud/panorama de aptitud (*fitness* / *fitness function* / *fitness landscape*)
Búsqueda de novedades (*novelty search*)
Clasificación (*ranking*)
Condición de terminación (*termination condition*)
Convergencia (*convergence*)
Convergencia prematura (*premature convergence*)
Cruce/operador de cruce (*crossover* / *crossover operator*)
Descendencia (*offspring*)
Elitismo (*elitism*)
Enfoque de fuerza bruta (*brute force approach*)
Escalada simple (ascenso de colinas o *hill climbing*)
Espacio del problema (*problem space*)
Estrategia de selección (*selection strategy*)
Exploración versus explotación (*exploration* vs *exploitation*)
Extremos locales (*local extrema*)
Grupo de apareamiento (*mating pool*)
Heurística (*heuristic*)
Intratabilidad computacional (*computational intractability*)
Muestreo universal estocástico (*stochastic universal sampling*)
Mutación/tasa de mutación (*mutation* / *mutation rate*)
Optimización (*optimization*)
Optimización combinatoria (*combinatorial optimization*)
Parámetros de inicialización (*initialization parameters*)
Población (*population*)
Recocido simulado (templado simulado o *simulated annealing*)
Recorrido (*tour*)
Selección de rueda de ruleta (*roulette wheel selection*)
Selección de torneo (*tournament selection*)
Selección de truncamiento (*truncation selection*)

---

**Fuentes:**