

# Programación paralela con algoritmo genético de optimización

Juan Pablo Bedoya Sánchez  
José Manuel González Grisales  
Jonatan Jair Leal González




# OBJETIVOS

## GENERAL

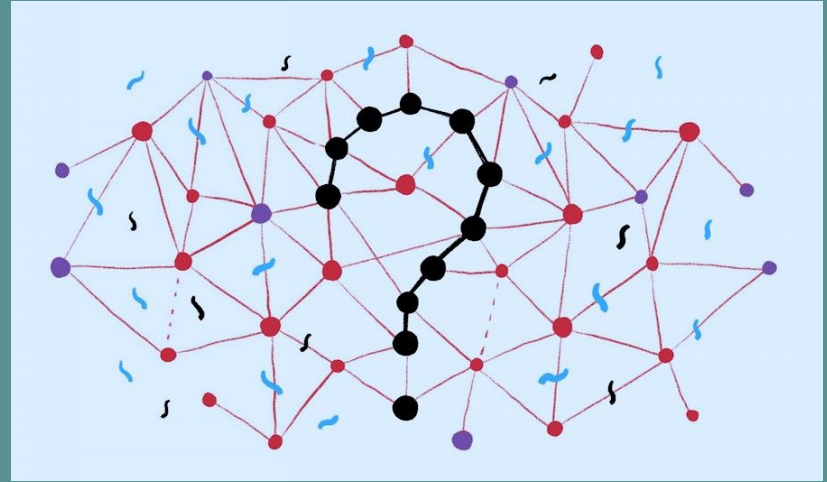
- Implementar y comparar un algoritmo de optimización genética tanto en su versión secuencial como paralela, evaluando su rendimiento en distintos lenguajes de programación.

## ESPECÍFICOS

- Desarrollar e implementar el algoritmo genético en diferentes lenguajes de programación, considerando tanto su versión secuencial como su versión paralela.
  - Experimentar con la paralelización del algoritmo utilizando distintas cantidades de núcleos y evaluar el impacto de esta configuración en el rendimiento.
  - Realizar una comparación detallada entre las versiones secuenciales y paralelas, midiendo métricas clave como tiempo de ejecución.
- 

# ALGORITMOS GENÉTICOS

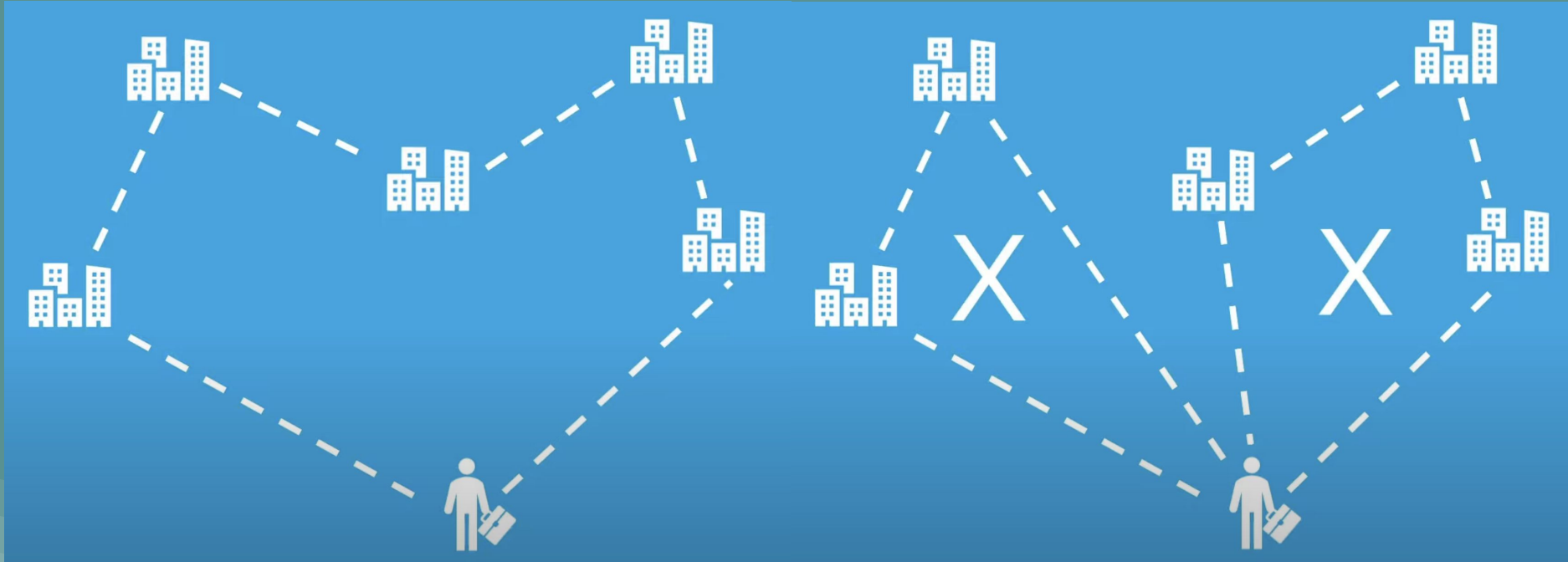
Un algoritmo genético (AG) es un método para solucionar problemas de optimización con o sin restricciones basándose en un proceso de selección natural que imita la evolución biológica. Este algoritmo modifica repetidamente una población de soluciones individuales. En cada paso, el algoritmo genético selecciona individuos de la población actual aleatoriamente y los utiliza como padres para producir los hijos de la siguiente generación. Tras varias generaciones sucesivas, la población "evoluciona" hacia una solución óptima



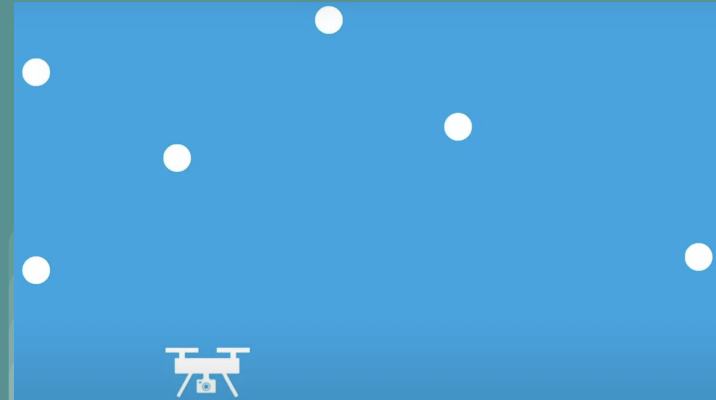
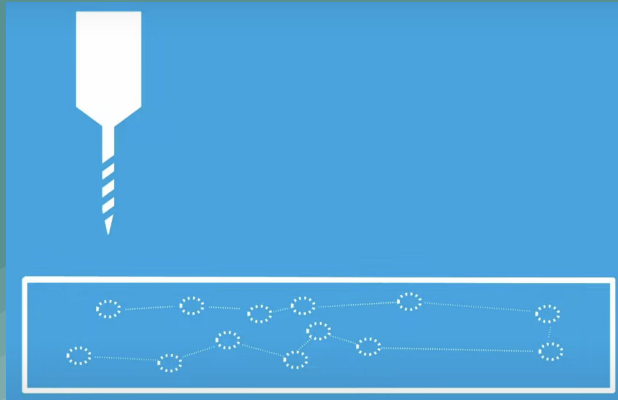
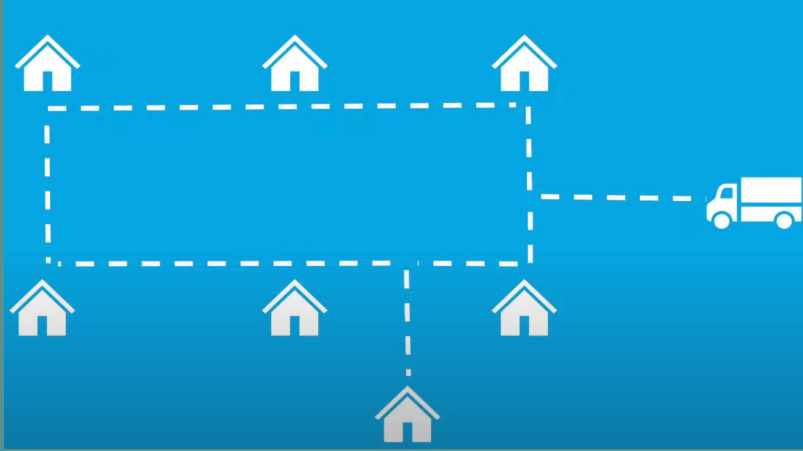
# TRAVEL SALESMAN PROBLEM

¿EN QUÉ ORDEN SE DEBE VISITAR LAS CIUDADES PARA HACER EL RECORRIDO MÁS CORTO, LUEGO RETORNANDO AL PUNTO DE ORIGEN?

Múltiples viajes no son permitidos\*

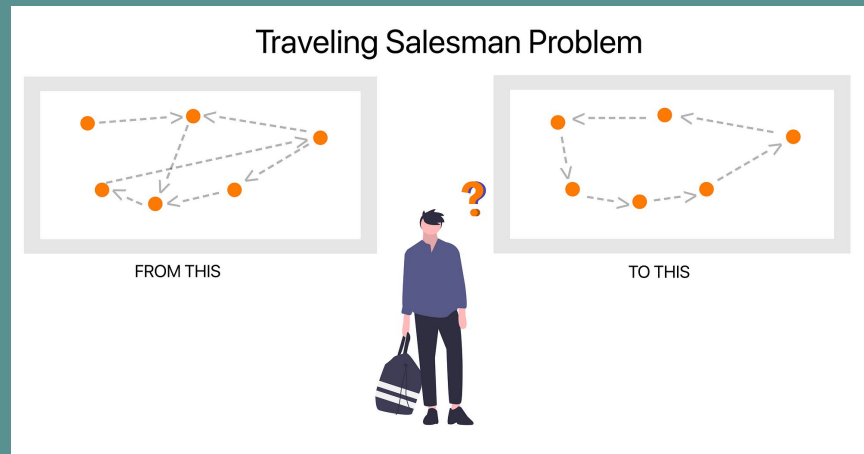


# OTRAS APLICACIONES



# EXPERIMENTO

1. Selección de variables: tiempo, ciudades (2, 4, 6, 8, 10). lenguajes (Java, Python).
2. Programación algoritmo en java y python, tanto versión secuencial como paralela.
3. Ejecutar experimento en misma máquina bajo misma condiciones
4. Tomar tiempos.
5. Almacenar resultados.
6. Graficar.
7. Sacar conclusiones.



# EXPERIMENTO

## Especificaciones de Hardware:

- RAM: 16 GB
- Procesador: i3 6100
- Sistema Operativo: Linux Mint
- Número de núcleos procesador: 2
- Número hilos procesador: 4

## Código:

[https://github.com/JoseMAGG/proyectos\\_2024-2/tree/main/GR-05/codigo](https://github.com/JoseMAGG/proyectos_2024-2/tree/main/GR-05/codigo)



# RESULTADOS

Java Toma 1		
Ciudades	Tiempo (S) Secuencial	Tiempo(S) Paralelo
2	0,012	0,006
4	0,011	0,009
6	0,019	0,011
8	0,042	0,066
10	0,825	1,722

Python Toma 1		
Ciudades	Tiempo (S) Secuencial	Tiempo(S) Paralelo
2	0,000093	0,019
4	0,000021	0,02
6	0,0004315	0,022
8	0,0365767	0,0783
10	3,1185	6,2666



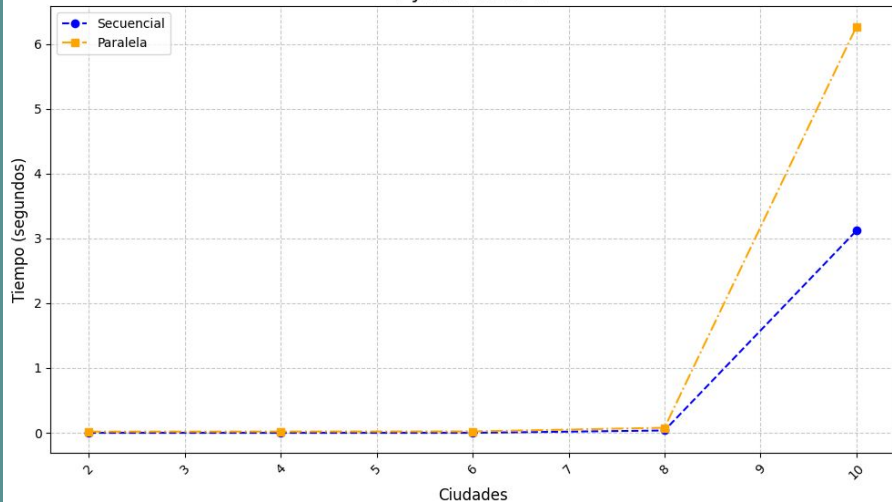
# RESULTADOS

Java Toma 2		
Ciudades	Tiempo (S) Secuencial	Tiempo(S) Paralelo
2	0,01	0,006
4	0,012	0,009
6	0,016	0,011
8	0,044	0,069
10	0,832	1,864

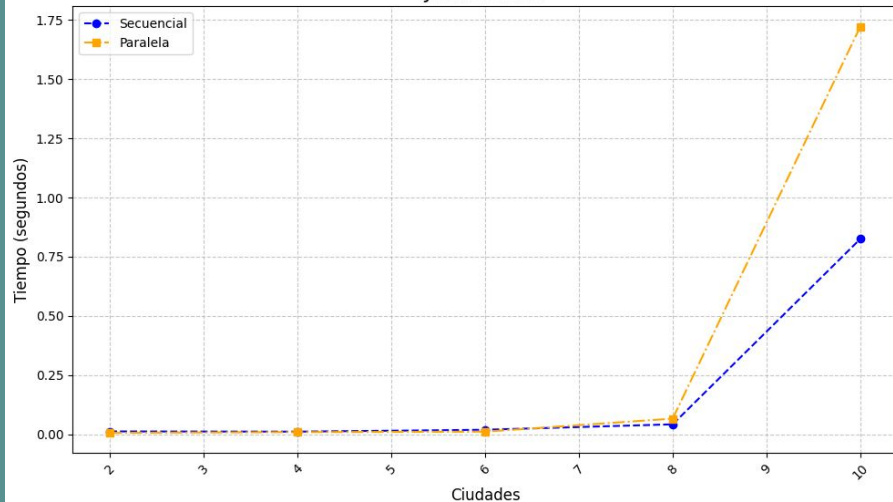
Python Toma 2		
Ciudades	Tiempo (S) Secuencial	Tiempo(S) Paralelo
2	0,0000107	0,021
4	0,00002	0,02
6	0,0007591	0,023
8	0,0388274	0,0726
10	3,0672	5,9634

# COMPARACIÓN PYTHON VS JAVA

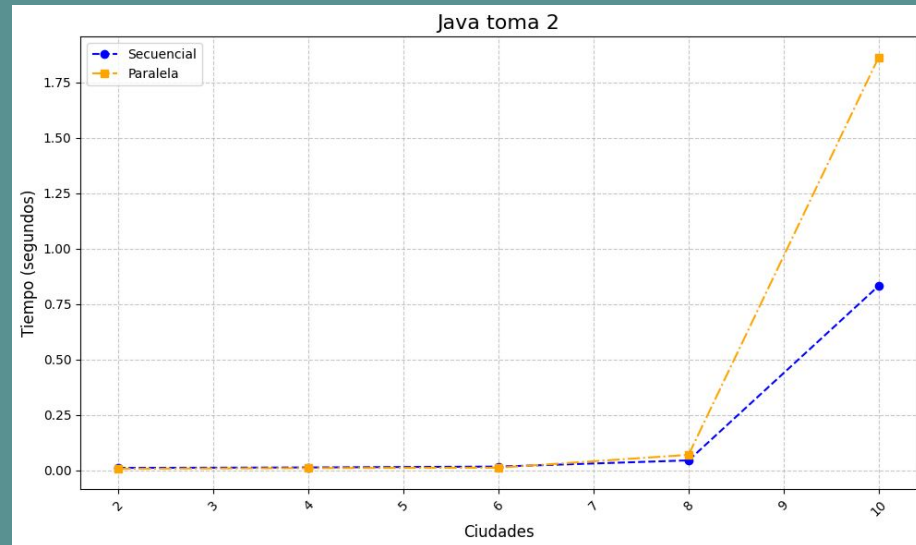
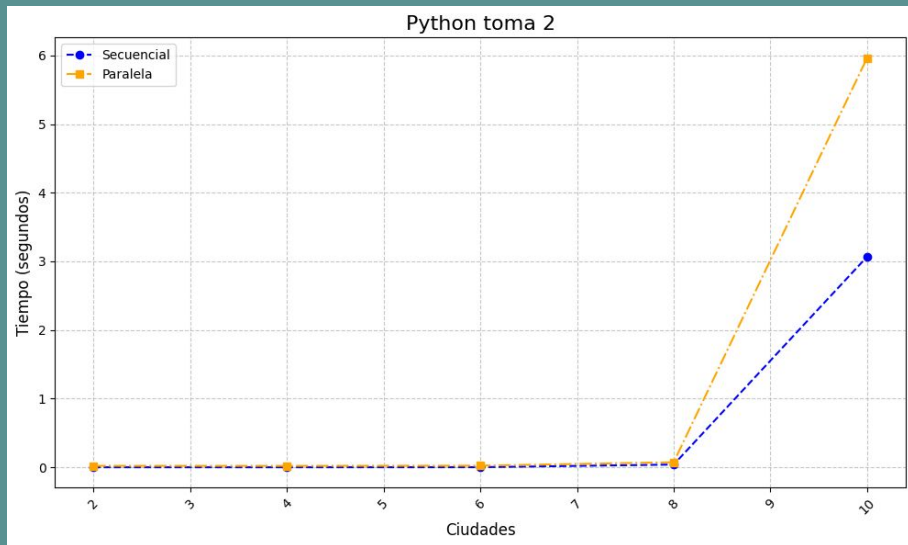
Python toma 1



Java toma 1



# COMPARACIÓN PYTHON VS JAVA



# CONCLUSIONES

1. En ambas gráficas (Java y Python), el tiempo de ejecución incrementa de manera exponencial a medida que aumenta el número de ciudades, particularmente a partir de 8 ciudades.
2. Para el mismo número de ciudades, los tiempos en Python (especialmente para la ejecución paralela) son considerablemente mayores que los de Java.
3. En ambas implementaciones, el paralelismo no parece aportar una ventaja significativa en términos de tiempo para resolver el problema en instancias pequeñas y medianas. De hecho, resulta menos eficiente para casos grandes.
4. Para resolver este problema, Java parece ser más eficiente que Python en este caso específico.

# BIBLIOGRAFÍA

1. “Algoritmo genético”. MathWorks - Creador de MATLAB y Simulink - MATLAB y Simulink - MATLAB & Simulink. Accedido el 16 de octubre de 2024. [En línea]. Disponible: <https://es.mathworks.com/discovery/genetic-algorithm.html>
2. La universidad en internet. “La computación paralela: características, tipos y usos”. UNIR. Accedido el 16 de octubre de 2024. [En línea]. Disponible: <https://www.unir.net/revista/ingenieria/computacion-paralela/>
3. ¿Qué es el Travelling Salesman Problem (TSP) y cómo solucionarlo? (s/f). Simpliroute. Recuperado el 1 de diciembre de 2024, de <https://simpliroute.com/es/blog/que-es-el-travelling-salesman-problem-tsp-y-como-solucionarlo>
4. AlphaOpt [@alphaopt2024]. (s/f). What is the Traveling Salesman Problem? Youtube. Recuperado el 1 de diciembre de 2024, de <https://www.youtube.com/watch?v=1pmBjIZ20pE>
5. Vargas, E. M., Martínez, M. R., Castillo, L. R. M., & Solis, L. S. (n.d.). Implementación paralela de un algoritmo genético para el problema del agente viajero usando OpenMP. Ipn.Mx. Retrieved December 2, 2024, from [https://rcs.cic.ipn.mx/2016\\_128/Implementacion%20paralela%20de%20un%20algoritmo%20genetico%20para%20el%20problema%20del%20agente%20viajero.pdf](https://rcs.cic.ipn.mx/2016_128/Implementacion%20paralela%20de%20un%20algoritmo%20genetico%20para%20el%20problema%20del%20agente%20viajero.pdf)

**¡GRACIAS!**

