Closest Pair Algorithm*

Katy Díaz Beltrán and Tabata Llach Brugés
Departamento de Ingenieria de Sistemas
Universidad del Norte
Barranquilla, Colombia
jdiazk@uninorte.edu.co

***Abstract*—This article is made to do the analysis of the closest pair algorithm using our won implementation and see if the time complexity differs if we use the tools given by the language and if differs, why this happened.**

***Index Terms*—Algorithm, time complexity, closest pair.**

INTRODUCCIÓN

Un algoritmo es un "conjunto ordenado y finito de operaciones que permite hallar la solución de un problema", al implementarlos hacer un analisis detallados del comportamiento de los mismos es indispensable para conocer el tiempo de complejidad y cuales son los casos que este produce para saber si el algoritmo es eficiente. El estudio de los algoritmos y el uso de las herramientas adquiridas por el estudio de ingenieria deberia ayudar a crear mejores algoritmos, mas eficientes.

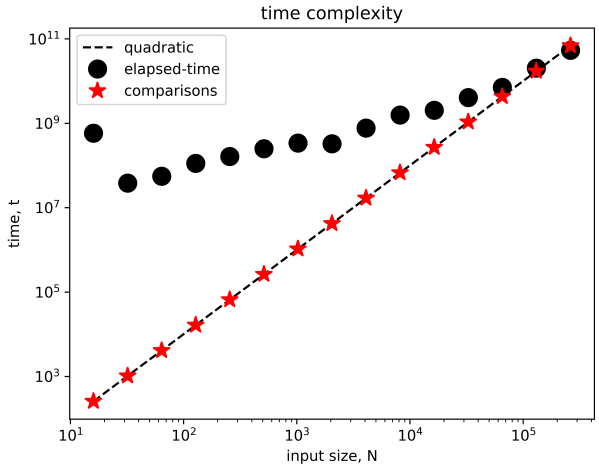
DEFINICIÓN DEL PROBLEMA

Closest pair es un algoritmo encargado de hallar el par mas cercano, para esto se tuvo en cuenta un dato de entrada, el cual permite que este funcione, este es importante ya que con algoritmos como este se pueden solucionar problemas de nuestro día a día asi como tambien se pueden realizar estudios de algoritmos en la ingenieria. La implementacion de este algoritmo sera implementando la estructura de dato Linked List y el nodo usado por esta lista.

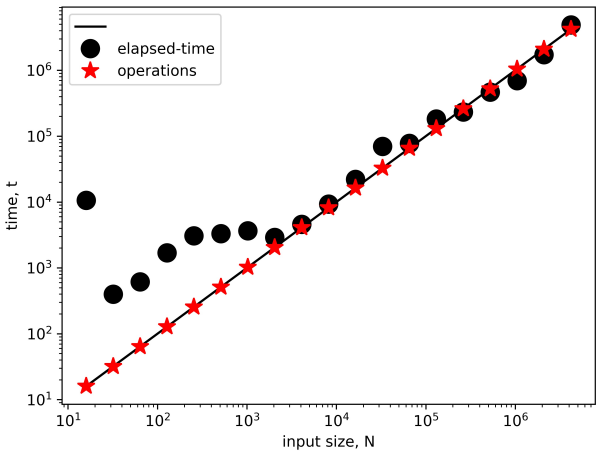
METODOLOGÍA

El algoritmo se realizó por medio de un software especializado llamado *IntelliJ IDEA*.
Las graficas fueron hechas en *Visual Studio Code* usando *Python*.
Para comprobar la hipótesis inicial del tiempo de complejidad se ejecutó el programa con diferentes tamaños.
Con tiempo de ejecución hacemos referencia en el tiempo que tarda el algoritmo en encontrar las coordenadas mas cercanas. Para mayor precision de este experimento se deberia usar varios hardware para comparar las diferencias del tiempo dependiendo de las condiciones del equipo. Ahora, el algoritmo es el siguiente:
El programa divide a la mitad la lista de coordanadas, organizando los datos por X primariamente y secundariamente por Y. Despues de ello el algoritmo verifica que el numero de coordenadas en cada particion sea menor a 3 para usar el algoritmo implementado *BrutalForce*, de no serlo, seguira haciendo las particiones de forma recursiva.
La diferencia entre este laboratorio y el anterior es que en este se implementa LinkedList en vez de ArrayList dado por Java.

RESULTADOS



Closest Pair using our own LinkedList



Closest Pair using ArrayList

Como resultado se obtuvo que, mas o menos, el time complexity del algoritmo es cuadratico. Este comportamiento se observa en valores de N muy grandez. Ahora, Los valores de N muy pequeños el tiempo de ejecucion es mucho mayor al esperado, esto se podria producir debido que la computadora

esta usando recurso de otro lado y por eso la demora, el porque el algoritmo empieza a tener un comportamiento cercano al esperado habria que analizar las coordenadas generadas. Ahora, podemos analizar que el algoritmo es mas eficiento la usar ArrayList, o sea la usar herramientas que el lenguaje nos ofrecen. Para futuras invetigaciones se podria revisar las funciones implementadas en LinkedList para poder mejorarlas y asi afectar positivamente al time complexity

CONCLUSIONES

Es mejor, hablando de eficiencia, usar las estrucuturas de datos ofrecidas por el lenguaje que nuestra implementacion. De todos modos, resaltamos la importancia de estos ejercicios para el entendiminto de como una implementacion afecta el rendimiento general del algoritmo, ademas de servir para seguir trabajnod la logica del ingeniero de sistema a cargo de la tarea.

AGRADECIMIENTOS

Agradecemos al profesor Misael por su ayuda al desarrollo de este proceso, gracias a su explicacion y teoria aprendida en clases, pudimos aplicarla en el analisis anterior.

REFERENCES

[1] Graphic creator code: . Diaz Maldonado, M. loglogplot.py. https://github.com/misael-diaz/computer-programming/blob/main/src/io/java/loglogPlot.