Instituto Politécnico Nacional
Escuela Superior de Cómputo

Análisis de Algoritmos, Sem: 2022-2, 3CV11, Práctica 6, 07 de junio de 2022

Práctica 6: Programación Dinámica

Luis Francisco Renteria Cedillo, Denzel Omar Vazquez Perez.

*lrenteriac*1400@*alumno.ipn.mx*, *dvazquezp*1600@*alumno.ipn.mx*

Resumen: En el presente documento se muestra la aplicación del algoritmo "Subsecucia Común mas Larga" con el objetivo de comparar dos archivos de código fuente en C y obtener un porcentaje de puntuación de similitud.

Palabras Clave: Antiplagio, Programación Dinámica, Expresiones Regulares, Python

1 Introducción

Se entiende como plagio el copiar, imitar y/o atribuirse una obra que no es de nuestra autoría, violando uno de los derechos morales el cual es el derecho patrimonial de la obra. En términos académicos, el hacer plagio es considerado como una falta ética y en la mayoría de los casos es fuertemente sancionado, ya que conlleva la intención de engañar a una comunidad y robar una obra ajena para obtener un beneficio o ventaja. Sin embargo, existe el derecho de citar al autor a pequeñas partes de una obra, siempre y cuando no se considere una reproducción sustancial de la misma.

Una herramienta popular en universidades e institutos es el software Turnitin, el cual es un servicio de prevención de plagio, utilizado en mas de 140 países y alrededor de 15,000 instituciones. Dicha herramienta fue creada a finales de los años 90 por estudiantes de la Universidad de California en Berkley. Su objetivo principal fue el facilitar la corrección por pares y en 2002 fue constituido Turnitin como se conoce hoy en día.

El funcionamiento básico de Turnitin es el siguiente: el usuario manda un archivo de texto al servicio de Turnitin, el cual se procesa y se calcula un porcentaje de similitud respecto a la base de datos interna de Turnitin. De esta manera, los profesores pueden revisar los trabajos de los alumnos de manera automatizada y decidir si una obra es plagiada o no.

2 Conceptos Básicos

2.1 Expresiones Regulares

En teoría de la computación, una expresión regular se utiliza para realizar una búsqueda en una cadena de texto conforme a un patrón, empleando una combinación de caracteres. Dicho de otra forma, las expresiones regulares brindan una manera muy flexible para reconocer cadenas de texto. Existen tres operadores que se utilizan en la construcción de las expresiones regulares: unión, concatenación y cerradura de Kleene.

2.2 Programación dinámica

La programación dinámica es un enfoque algorítmico para buscar la solución de un problema de optimización dividiéndolo en varios subproblemas más simples, observando que el problema global depende de la solución óptima de sus subproblemas. Por lo tanto, la característica más esencial de programación dinámica es la estructura adecuada de los problemas de optimización en múltiples niveles, que se resuelven secuencialmente a un nivel a la vez.
Mediante el uso de técnicas ordinarias de problemas de optimización, se resuelve cada nivel y su solución ayuda a definir las características del problema del siguiente nivel en la secuencia. Comúnmente, los niveles representan diferentes periodos de tiempo en la perspectiva del problema general.

2.3 Algoritmo de la Subsecucia Común mas Larga

Algoritmo es usado para encontrar la subsecuencia comń más larga entre dos cadenas de caracteres. Dicha subsecuencia no necesariamente tiene que ser contigua en ambas cadenas.

3 Experimentación y Resultados

3.1 Análisis a Priori

Dada la implementación del algoritmo de la subsecucia común mas larga en el lenguaje de programación Python se muestra en la Figura 1, la codificación de este.
Entonces sea cada bloque de secuencias del código, se determina el orden de complejidad del para el peor caso de este por medio por medio del análisis de segmentos de código concluyendo que el algoritmo tiene orden de complejidad $T(n) \in \Theta(mn)$.

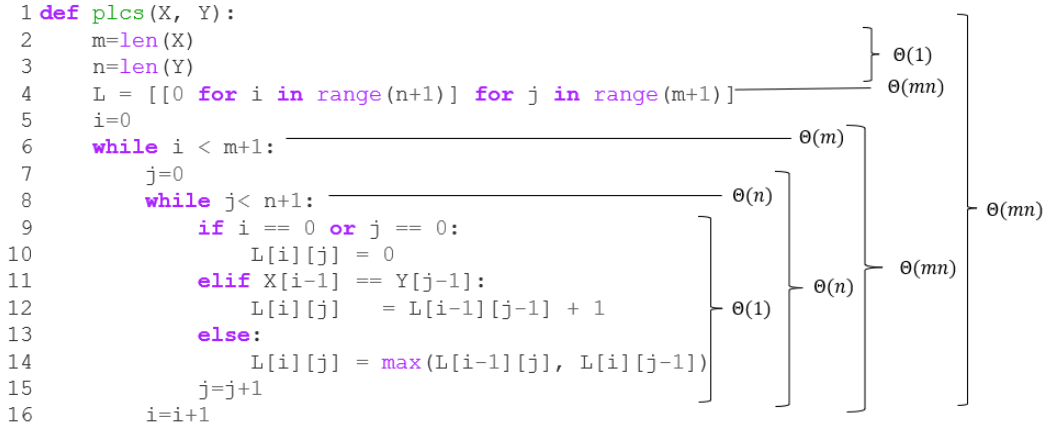


Figura 1: Análisis por bloques de código del algoritmo

3.2 Resultados

Con ayuda del algoritmo de la subsecucia común mas larga, se pone en práctica en un programa que comparar dos archivos de código fuente en lenguaje C con el propósito de detectar el porcentaje de coincidencias que puedan tener.

A continiación se muestran los resultados a partir de archivos en los que solo se hizo cambios de nombres a variables asi como códigos totalmente distintos.

Para el primer caso se escogió el archivo *NQueens.c*, con el objetivo de comparar el mismo archivo y obtener el 100% de las coincidencias puesto que se analiza el mismo código fuente, así en la Figura 2 se aprecia que se logro obtener lo esperado.

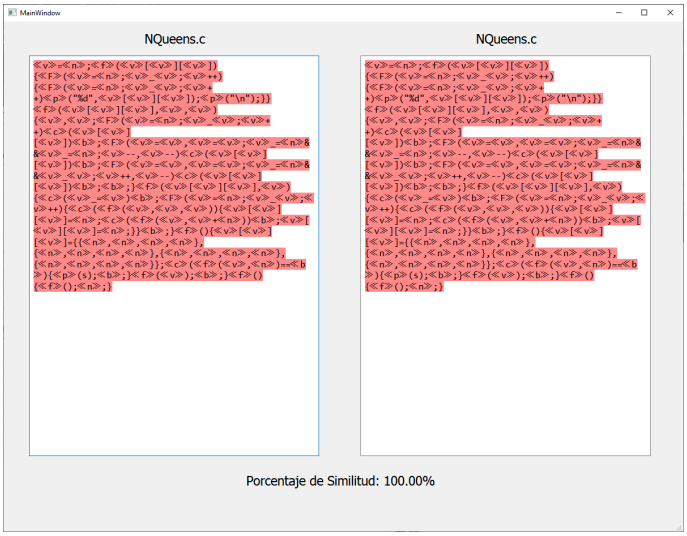


Figura 2: Prueba con el mismo archivo de código fuente

El siguiente caso se selecciono y modifico haciendo el cambio al nombre de las variables de *NQueens.c*, este archivo se nombro *NQueensPlagio.c*, así en la Figura 3 el programa implementado detecta que el programa sigue siendo el mismo aun cambiando el nombre de la variables del código fuente, por tanto el porcentaje de coincidencia es del 100%.

Figura 3: Prueba con el cambio del nombre de las variables

Para el tercer y ultimo caso, se hace uso de dos archivos de código fuente totalmente distintos, para verificar que el porcentaje de coincidencia sea inferior

existiendo coincidencias mínimas entre estos. Asi en la Figura 4 se muestra que los archivos *heap_sort.c* y *quick.c* tienen un porcentaje de coincidencia del 18.38%.

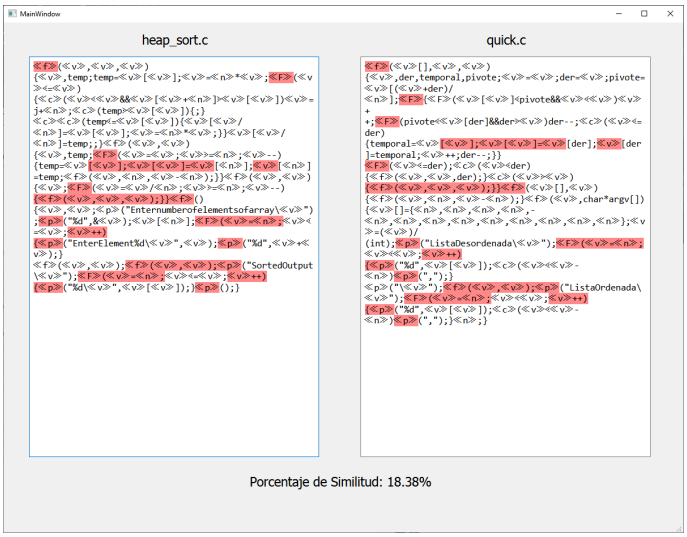


Figura 4: Primer prueba con distintos archivos

Para Figura 5 se muestra el segundo ejemplo donde los archivos *merge.c* y *quick.c* son diferentes y presentan un porcentaje de coincidencia del 14.56%.

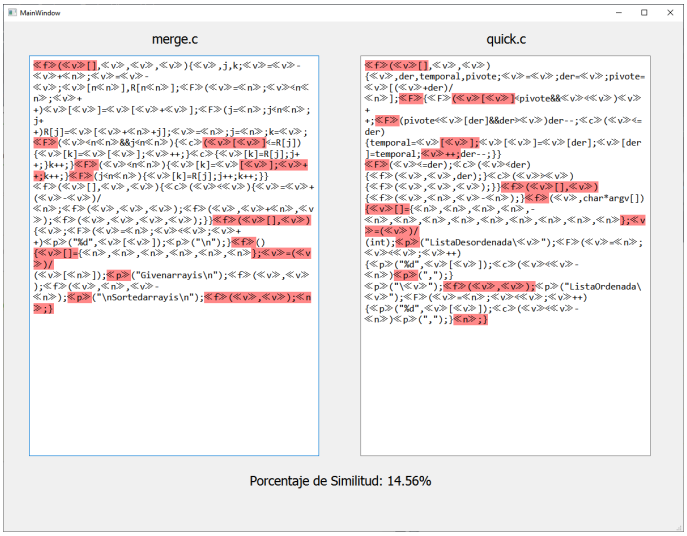


Figura 5: Segunda prueba con distintos archivos

Finalmente en la Figura 6 se muestra el tercer ejemplo que hace uso de un

archivo extraído de internet y uno en el que se trabajo , *dijkstra.c* y *dijkstra2.c* respectivamente, teniendo un porcentaje de coincidencia del 22.78%.

Figura 6: Tercer prueba con distintos archivos

El programa resulta eficaz, puesto al tokenizar los archivos con los que se trabaja es posible cambiar las palabras reservadas de lenguaje C usando literales que denotan el tipo de sentencia que se esta manejando en el instante gracias a expresiones regulares, así al finalizar cambio antes comentado se implementa el algoritmo de la subsecuencia común mas larga, mostrando las coincidencias encontradas marcadas con color rojo en el proceso. Así se determina que para detectar el plagio es necesario encontrar la frase mas larga compartida, dado que las frases son cadenas de caracteres consecutivos y aquí se necesita la subsecuencia común más larga entre los códigos fuente.

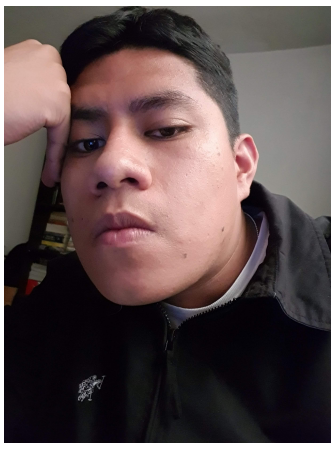
4 Conclusiones

Luis Francisco Renteria Cedillo

Cuando implementamos la búsqueda de la subsecuencia común más larga, en su primer versión, nos percatamos que el porcentaje era muy alto aún cuando los códigos eran totalmente distintos, esto se debe a que en nuestro código, en su versión inicial, identificaba cada carácter como un elemento, por tal motivo, cambiamos la implementación utilizando los elementos de entrada como cadenas de sentencias simples. Con este nuevo enfoque, nuestra búsqueda ahora era entre listas de cadenas, y como se mostró en los resultados, el porcentaje es el esperado conforme a cada caso.

Además, se optó por agregar una simple interfaz gráfica para poder observar los elementos que son similares en cada código fuente. Finalmente se concluye que el programa logra su objetivo al calcular porcentajes de puntuación de similitud.

Denzel Omar Vazquez Perez



En la realización de la práctica se percato que el problema de la subsecuencia común mś larga surge cada vez que buscamos similitudes en diferentes textos como lo son los archivos fuentes de de código de un programa, gracias a conocer la estrategia de programación dinámica permite resolver el problema en subproblemas almacenando soluciones antes encontradas para dar por ultimo la solucióon óptima global.

También, durante la investigación para la implementación del algoritmo en le programa de plagio, se vio que otra aplicación interesante es encontrar un consenso entre las secuencias biológicas como si se hablara de arreglos de caracteres, y esto es por que los genes para construir las proteínas se transforman con el tiempo, pero las regiones funcionales permanecen constantes para que funcionen correctamente, por tanto se concluye que las subsecuencia común más largo del mismo gen en diferentes especies proporciona información sobre lo que se ha conservado a lo largo del tiempo.

5 Bibliografía

Brassard, G. (1997). *Fundamentos de Algoritmia*. España: Ed. Prentice Hall.

Cormen, E. A. (2022). *Introduction To Algorithms*, 3Rd Ed. Phi.

Nayak S. (2020). *Fundamentals of Optimization Techniques with Algorithms*. Elsevier.

Python Software Foundation (2021) *re: Operaciones con Expresiones Regulares* Dispoible en: https://docs.python.org/es/3/library/re.html