Imagen que contiene firmar

Descripción generada automáticamente

Análisis y Diseño de Algoritmos Avanzados

12 de septiembre de 2021

Carlos Moises Chavez Jimenez

ITESM

A01637322

Actividad Integradora 1: Reflexión

**Reflexión**

Para empezar mi reflexión, me gustaría decir que me parece muy practico el haber aprendido nuevas formas de atacar estos problemas de cadenas. Ya que de la manera que yo los realizaba, estos tenían complejidades grandes y no parecían ser la mejor opción.

Para realizar la solución de toda la situación problema 1, convertimos el contenido de los archivos de texto en una cadena por cada archivo. Los saltos de línea los eliminamos y solamente dejamos el contenido con el que vamos a trabajar. La complejidad es O(n) donde n es el número de caracteres del archivo.

El algoritmo que nos ayudó para realizar el primer problema fue el de Knuth-Morris-Pratt. Este es un algoritmo muy práctico que hace más eficiente la búsqueda de una cadena dentro de otra. Esto lo hace a través de una tabla de fallo, que no permite que se analice un carácter más de una vez. A diferencia de la fuerza bruta, que procesa cada carácter un gran número de veces (dependiendo del número de comparaciones). Esto nos permitió facilitarnos mucho el trabajo, y hacer más eficiente la solución del primer ejercicio. La complejidad es O (n + m) donde n y m son los caracteres de ambas cadenas que se recibieron de parámetro.

Para el segundo ejercicio utilizamos el algoritmo manacher, que es muy efectivo para encontrar palíndromos. Este inserta un carácter que sabemos que no está en la cadena (en nuestro caso #) dentro de la cadena original. Lo inserta al principio, al final, y entre cada carácter subyacente. Con eso resuelve la parte de considerar cadenas de palíndromos pares e impares. Y con la ayuda de un vector auxiliar, comenzamos a guardar las longitudes de los palíndromos que encontremos. Para al final devolver la más grande guardada dentro del vector. Con este ejercicio tuvimos unos problemas con el pre y post procesamiento. Que son las funciones que agregan y quitan los caracteres de apoyo en la cadena original. Pero lo solucionamos con un vector auxiliar que guardaba todo lo que no fuera el carácter especial. Su complejidad O (n) donde n es el tamaño de la cadena entera.

Por último, para el tercer ejercicio utilizamos un algoritmo de programación dinámica con el uso de una suffix. Esta matriz se hace para poder guardar los tamaños de los suffixes de las sub-cadenas más largas. Las guías de filas y columnas nos van a permitir guardar el índice de la cadena mayor a todas. Y así, al final, poder recorrer la cadena con ese índice de derecha a izquierda. Desde hace tiempo que conocíamos los preffixes y suffixes, pero en esta situación es donde pudimos ver su uso real. Aquí fue donde conocimos su importancia y eficacia para la resolución de este problema. Su complejidad: O (m\*n) donde m y n son las longitudes de ambos archivos de transmisión.

En general, esta actividad se nos hizo más complicada que las anteriores por el tener que aprender nuevos algoritmos para resolver estos problemas. Pero esto nos ayudó en cierta parte, porque repasamos las presentaciones vistas en clase y nos permitió entender mejor todo lo visto en clase con relación a las cadenas. Lo que también nos parece bueno es que, al ya conocer bien estos algoritmos, en un futuro si se nos presentan problemas que requieran la misma lógica y análisis, ya sabemos qué es lo que debemos implementar y cómo.