Tecnológico de Monterrey
Escuela de Ingeniería y Ciencias

Campus Monterrey

Nombre del trabajo:

Actividad Integradora 1

Curso:

Análisis y diseño de algoritmos avanzados (Gpo 607)

Alumno:

Julian Lawrence Gil Soares

Matrícula:

A00832272

Fecha de entrega:

29 de noviembre

Nuestro proyecto integrador consistió en implementar un programa que pudiera analizar los datos contenidos en 2 archivos de texto y compararlos con 3 archivos conteniendo caracteres que en nuestro caso representan código malicioso. Un reto que tuvimos que enfrentar durante este proyecto es que por el tamaño de los archivos de entrada fue crucial que nuestro programa fuera lo más optimo posible.

La primera parte del reto fue revisar los archivos de transmisión y buscar si existía alguna ocurrencia de los archivos maliciosos. Para esto empleamos el algoritmo KMP este algoritmo es una de las maneras óptimas para resolver este tipo de problemas ya que usando el patrón a buscar puede reducir el número de comparaciones necesarias para encontrar el patrón dentro del texto que se compara. Este algoritmo tiene una complejidad de O(n).

El siguiente reto que enfrentamos fue el de resolver el problema de buscar y encontrar el palíndromo más largo dentro del archivo para esto tuvimos un problema encontrando una manera de resolverlo en menos de O(n^3) a través del método de fuerza bruta. Terminamos encontrando el algoritmo de manacher que lo resuelve en tiempo lineal ósea O(n). Este algoritmo funciona buscando el carácter que más se puede expandir en un palíndromo, calcula el palíndromo más largo presente usando información de los caracteres previos del string. Primero compara los caracteres que están a lado de cada carácter y le asigna un valor en base a la longitud que se podría formar usando lo como un punto medio del palíndromo por ejemplo el primer carácter no tiene un carácter a su lado así que tiene un valor de 0 pero digamos que el segundo carácter tiene dos letras iguales a su lado entonces tiene un valor de 1 usando este valores podemos determinar que el carácter que le sigue puede ser el punto medio de un palíndromo de longitud 3 gracias a la propiedad de simetría de los palíndromos, este proceso lo repetimos hasta tener un valor asignado para cada carácter. Después revisamos desde el punto medio que puede tener el palíndromo más largo los caracteres a su alrededor hasta encontrar el palíndromo más largo presente en el string.

Longest comon substring crea una matriz bidimensional donde guarda los substrings presentes en ambos textos después compara los substrings con la misma longitud de mayor a menor hasta encontrar el mayor substring común presente en ambos. Esta solución tiene una complejidad de O(n*m)