Tecnológico de Monterrey

Curso:

Análisis y diseño de algoritmos avanzados

Grupo:

570

Estudiantes:

Santiago Arista Viramontes - A01028372

Diego Vergara Hernández - A01425660

José Leobardo Navarro Márquez - A91541324

Profesor:

Nezih Nieto Gutiérrez

Título:

E1. Actividad Integradora 1

Fecha de entrega:

26 de Enero del 2026

Resumen

En este documento de reflexión presentamos el diseño e implementación de nuestra solución para la Actividad Integradora 1, para el análisis de transmisiones de datos representadas en formato hexadecimal. El objetivo principal es detectar la presencia de código malicioso dentro de dichas transmisiones mediante la aplicación de algoritmos clásicos de búsqueda de patrones, análisis de palíndromos, comparación de similitud entre cadenas y técnicas de compresión estadística. La solución integra los algoritmos Knuth–Morris–Pratt, Manacher, programación dinámica para Longest Common Substring y codificación de Huffman, evaluando tanto su uso como su complejidad computacional.

Introducción

La detección de patrones maliciosos dentro de transmisiones requiere algoritmos que sean adecuados y escalables. En este proyecto se aborda dicha problemática utilizando una combinación de algoritmos de tiempo lineal y programación dinámica.

Las transmisiones y códigos maliciosos se representan como cadenas de caracteres hexadecimales (0–9, A–F). A partir de estas cadenas, se realizan cuatro análisis principales: búsqueda de subsecuencias, detección de palíndromos, comparación de similitud entre transmisiones y análisis estadístico mediante Huffman Coding.

Parte 1 - Búsqueda de Subsecuencias

Para determinar si un código malicioso aparece dentro de una transmisión utilizamos el algoritmo Knuth–Morris–Pratt (KMP). Este algoritmo permite buscar un patrón dentro de un texto sin retroceder de forma innecesaria, utilizando una prefix function que captura la estructura interna del patrón.

Complejidad

- Tiempo: $\mathbf{O(n+m)}$, donde n es el tamaño de la transmisión y m el tamaño del patrón.
- Espacio: $\mathbf{O(m)}$ para el arreglo de prefijos.

Esta complejidad es adecuada para archivos de gran tamaño, como los que serán utilizados al poner a prueba nuestra solución.

Parte 2 - Búsqueda del Palíndromos

Dado que el código malicioso puede encontrarse “mirrored”, decidimos usar el algoritmo de Manacher para identificar el palíndromo más largo en cada transmisión. Este algoritmo transforma la cadena original para unificar el tratamiento de palíndromos pares e impares y calcula el radio máximo alrededor de cada ‘centro’ posible.

Complejidad

- Tiempo: **O(*n*)**
- Espacio: **O(*n*)**

El uso de Manacher garantiza una solución óptima frente a enfoques con una complejidad cuadrática.

Parte 3 - Substring Común Más Largo

Para analizar la similitud entre ambas transmisiones, se calcula el Longest Common Substring (LCS). A diferencia del Longest Common Subsequence, este problema requiere que los caracteres sean contiguos. Utilizamos dynamic programming clásico en nuestra solución.

Complejidad:

- Tiempo: **O(*n* . *m*)**
- Espacio: **O(*n* . *m*)**

Aunque esta complejidad puede llegar a ser costosa para strings extremadamente grandes, consideramos que es aceptable dentro del contexto de esta actividad, permite obtener directamente las posiciones del substring común más largo.

Parte 4 - Huffman Coding

En la última parte se introduce la codificación de Huffman como una herramienta para mejorar la eficiencia y realizar un análisis estadístico de las transmisiones. Para cada archivo de transmisión se construye un árbol de Huffman basado en la frecuencia real de sus símbolos.

A partir de este árbol se calcula la longitud promedio esperada de codificación (bits per character), y el tamaño comprimido real de la transmisión.

Posteriormente, cada código malicioso se codifica utilizando el árbol de la transmisión correspondiente. Si la longitud codificada de un mcode es mayor que la esperada (usamos un umbral del 150%), este se marca como sospechoso.

Complejidad

- En la construcción del árbol: **O($k \log k$)**, donde k es el número de símbolos distintos.
- En la codificación: **O(n)**

Justificación del Uso de Huffman

El uso de Huffman Coding es muy efectivo cuando las transmisiones tienen una distribución sesgada de símbolos, como ocurre en los ejemplos proporcionados, donde ciertos caracteres (A, B, C y D) son mucho más frecuentes. Esto permite asignar códigos cortos a los símbolos dominantes y códigos largos a los que son menos frecuentes.

Cuando un mcode contiene símbolos raros respecto a la transmisión, su longitud aumenta de manera notable, lo que sirve como un indicador de anomalía. Huffman mejora la eficiencia de representación y además aporta un criterio adicional para detectar código malicioso.

Discusión y Resultados

Con el conjunto de pruebas proporcionado, observamos que:

- Los algoritmos de búsqueda identifican correctamente los patrones maliciosos y los palíndromos largos.
- El substring común más largo es detectado incluso cuando se encuentra rodeado de ruido.
- Huffman Coding da evidencia para notar diferencias claras entre datos típicos de la transmisión y anomalías en secuencias.

Conclusiones

Integramos múltiples algoritmos eficientes para resolver un problema realista de análisis de datos. Cada técnica fue seleccionada por su adecuación su respectivo problema, así como su complejidad. En particular, la incorporación de Huffman Coding mejora la eficiencia en tiempo y espacio, y también introduce un enfoque más estadístico, que es útil para la detección de anomalías. La solución final es modular, escalable y cumple con todos los requisitos de la rúbrica, demostrando la aplicabilidad práctica de los algoritmos que estudiamos en clase.