Instituto Politécnico Nacional

Escuela Superior de Computo

Unidad de Aprendizaje:

Análisis y Diseño de Algoritmos

Practica 6: “Codificación de Huffman”

Profesor: García Floriano Andrés

Alumnos:

López Martínez Jonathan
Mejia Mejia Mauricio Xavier
Nava Montiel Erasmo
Silva Vázquez Diego
Daniel Solis Bustos

Grupo: 3CV5

El presente programa implementa un sistema de compresión de texto basado en el algoritmo de codificación de Huffman, una de las técnicas más utilizadas para la compresión sin pérdida de información. El objetivo principal es analizar un texto ingresado por el usuario, calcular la frecuencia de cada carácter, construir un árbol de Huffman a partir de estos valores y, finalmente, generar un conjunto de códigos binarios optimizados que permitan representar el texto original utilizando una menor cantidad de bits. El proyecto está organizado modularmente en distintos archivos .c y .h, lo que permite mantener una estructura limpia, escalable y fácil de entender.

El archivo principal main.c es el encargado de coordinar todo el proceso. Primero solicita al usuario que escriba un texto, el cual se almacena en un arreglo de caracteres. Una vez leído el texto, el programa recorre todos sus caracteres y construye una tabla de frecuencias, en la cual cada posición del arreglo representa un carácter del código ASCII (256 posibles). Este conteo es esencial, ya que el algoritmo de Huffman se basa en la frecuencia de aparición para asignar códigos más cortos a los caracteres que ocurren con mayor frecuencia y códigos más largos a los menos comunes, logrando así una representación más eficiente.

Después del cálculo de frecuencias, se llama a la función imprimirFrecuencias (ubicada en utils.c), que muestra una tabla con cada carácter presente en el texto y su frecuencia correspondiente. Esto permite visualizar de manera clara cuáles son los símbolos que tendrán un peso mayor en la construcción del árbol.
El componente central del proyecto se encuentra en huffman.c, donde se definen las funciones necesarias para crear el árbol de Huffman. En este archivo se utiliza una estructura llamada Nodo, definida previamente en el header huffman.h, la cual contiene el símbolo representado, su frecuencia y punteros a sus nodos hijo izquierdo y derecho. La función crearNodo se encarga de reservar memoria para cada nodo y asignar sus valores iniciales.

El proceso de construcción del árbol inicia creando un nodo por cada carácter que tiene una frecuencia mayor a cero. Estos nodos se almacenan en un arreglo temporal. Posteriormente, mientras exista más de un nodo en el arreglo, se ordenan de acuerdo con su frecuencia utilizando la función ordenar. Después de ordenar, se toman los dos nodos con menor frecuencia y se combinan en un nuevo nodo interno, cuya frecuencia es la suma de ambas. El nodo izquierdo representa el bit '0' y el derecho el bit '1' en el código binario final. Este procedimiento se realiza repetidamente hasta que solo queda un nodo, el cual se convierte en la raíz del árbol de Huffman.
Durante este proceso, la función imprimirIteracion muestra el estado de los nodos en cada paso, lo que permite seguir visualmente la formación del árbol. Este enfoque es ideal para prácticas o entornos educativos, ya que facilita comprender la dinámica del algoritmo.

Una vez construido el árbol, se procede a generar los códigos de Huffman mediante la función recursiva generarCodigos. Esta función recorre el árbol desde la raíz, agregando un '0' al código cuando avanza hacia el subárbol izquierdo y un '1' cuando avanza hacia el derecho. Cada vez que se llega a un nodo hoja, el código formado hasta ese punto se almacena en la tabla codigos, donde cada posición corresponde al código binario del carácter asociado. Esto produce una representación única y eficiente para cada símbolo.

En main.c, después de generar los códigos, se imprimen en pantalla para que el usuario vea el código asignado a cada carácter presente en el texto. Posteriormente, con ayuda de la función codificarTexto del archivo utils.c, el programa recorre el texto original y muestra su versión codificada, sustituyendo cada carácter por su correspondiente secuencia de bits.

Finalmente, el programa calcula la tasa de compresión, es decir, el porcentaje de reducción en tamaño respecto al número original de bits. Esto se realiza mediante la función tasaCompresion, que compara el tamaño original del texto (considerando que cada carácter ocupa 8 bits) contra el tamaño total del texto codificado. Esta métrica permite evaluar de manera cuantitativa la efectividad de la codificación de Huffman para el texto ingresado.

En conjunto, el programa implementa de manera correcta y didáctica el algoritmo de Huffman, mostrando paso a paso la construcción del árbol, los códigos generados y los resultados de la compresión. La separación en módulos (main.c, huffman.c, utils.c y sus respectivos headers) facilita la comprensión del flujo y fortalece las buenas prácticas de programación estructurada en C, permitiendo mantener el código ordenado, reutilizable y fácil de extender.