Aplicaciones para comunicaciones de red, Sem: 2019-2, 3CM5, práctica 1.1, Fecha: 02/02/2019

Práctica 1.1: SERVICIO DE TRANSFERENCIA DE ARCHIVOS

Hernández Tapia Luis Enrique.

Escuela Superior de Cómputo
Instituto Politécnico Nacional, México
*tapia*641@*gmail.com*

Resumen: En esta práctica se desarrolla una plicación para seleccionar múltiples archivos y enviarlos a través del protocolo TCP
.

Palabras clave: TCP, IP, Sockets de flujo.

1 Introducción

El envío de archivos a través de la red es una característica importante para la gran mayoría de las aplicaciones que hoy día se utilizan (blogs, redes sociales, mensajería instantánea, declaración de impuestos, educación en línea, etc.), sin embargo, no todas las aplicaciones disponibles permiten el envío de archivos de gran tamaño (p.e. El correo electrónico no permite enviar archivos de más de 10 o 20 MB). Esto hace necesario el desarrollo de aplicaciones que permitan transferir archivos sin importar el tamao de éstos.

2 Conceptos Básicos

A continuación se presentan algunas definiciones para que al lector le sea comprensible el tema en cuestión.

Notación Θ: En todo lo que sigue consideramos funciones definidas sobre $\mathbb{N} \bigcup \{0\}$. Sea g(n) una función. Se define θ(g(n)) como:
θ(g(n))= $\{f(n) \mid \exists_n\ C_1, C_2 > 0$ y $n_o = 0$ tal que $0 \leq C_1 g(n) \leq f(n) \leq C_2 g(n)$ $\forall n \geq n_o\}$

Notación O: Dada una función g(n). Se define O(g(n)) como:
O(g(n))= $\{f(n) \mid \exists_n\ C_1 > 0$ y $n_o > 0$ tal que $0 \leq f(n) \leq C_1 g(n)\ \forall n \geq n_o\}$

Notación Ω: Dada una función g(n). Se define Ω(g(n)) como:
Ω(g(n))= $\{f(n) \mid \exists_n\ C_1 > 0$ y $n_o > 0$ tal que $0 \leq C_1 g(n) \leq f(n)\ \forall n \geq n_o\}$

Código de Huffman:
El algoritmo de Huffman es usado en la compresión de datos. Puesto es capaz de representar un conjunto de símbolos obtenidos a partir de cierto alfabeto, usando el menor número de bits posible, pero preservando en todo momento la capacidad de descomprimir o decodificar la información

El algoritmo se propuso en 1952 como una forma sencilla y óptima de mapear cada símbolo de un alfabeto con un código de longitud óptima. El proceso de asignación de códigos se lleva a cabo mediante la construcción de un árbol binario, desde las hojas hacia la raíz, de manera que los nodos hojas son los símbolos del alfabeto. A continuación un ejemplo gráfico.

Precedemos a comprimir la siguiente cadena de carácteres.

L	a	p	t	o	p

Enseguida se crea su tabla de frecuencias.

Crácter	Frecuencia
L	1
a	1
p	2
t	1
o	1

Table 1: Tabla de frecuencia

Ordenamos de forma ascendente con *QuickSort*

Crácter	Frecuencia
L	1
a	1
t	1
o	1
p	2

Table 2: Tabla de frecuencia ordenada

Insertamos en una lista enlazada de árboles los carácteres con su número de frecuencia. Tomamos los dos primeros nodos y creamos un árbol con ellos donde la raíz será la suma de la frecuencia de ambos. El nodo de la izquierda será el primero que tomamos y el de la derecha el segundo.

El resultado es: **11011110000110**

QuickSort:

Fue desarrollado por C.A.R. Hoare en 1960. El algoritmo original es recursivo, pero se utilizan versiones iterativas para mejorar su rendimiento (los algoritmos recursivos son en general ms lentos que los iterativos, y consumen ms recursos). El algoritmo fundamental es el siguiente:

- Eliges un elemento de la lista. Puede ser cualquiera. Lo llamaremos elemento de división.
- Buscas la posición que le corresponde en la lista ordenada.
- Acomodas los elementos de la lista a cada lado del elemento de división, de manera que a un lado queden todos los menores que él y al otro los mayores.
- En este momento el elemento de división separa la lista en dos sublistas.
- Realizas esto de para cada sublista mientras stas tengan un largo mayor que 1.

Estrategia Greedy:

A continuación se presenta la estrategia utilizada en la codificación de Huffman denominada como el método voraz o *Greedy Method*. El propósito de un algoritmo voraz es encontrar una solución, es decir, una asociación de valores a todas las variables tal que el valor de la función objetivo sea óptimo. Para ello sigue un proceso secuencial en el que cada paso toma una decisión .

La decisión es localmente óptima, es decir, ningún otro valor de los disponibles para esa variable lograría que la función objetivo tuviera un mejor valor.

El siguiente paso del algoritmo voraz es encontrar un problema idéntido, pero estrictamente menor, al que tenía en el paso anterior y vuelve a plicar la misma función de selección para tomar la siguiente decisión. Un criterio importante es nuncaintentar asignar un nuevo valor a la misma variabl, es decir, no regresarse.

Pseudocódigo:

3 Experimentación y Resultados

En la práctica se implementó el algoritmo de codificación de Huffman, sujeto a los siguientes criterios.

1. Implementar el algoritmo de la codificación de Huffman con las siguientes condiciones:

Para la Codificación.
Entrada: Un archivo de texto con extensión .txt (original.txt) a codificar.
Salida: se generaran tres archivos .txt.
Frecuencias.txt: Mostrará la tabla de frecuencias de los caracteres que aparecen en el archivo original.txt.

4 Conclusiones

Luis Enrique: La compresión de archivos la hemos venido manejando sin tener noción de algún algoritmo que permita ésto. Con la codificación de Huffman tenemos una estrategia Greedy para realizarlo, debido a que llega a una solución local, obteniendo la tabla de frecuencias y para obser el resultado de la compresión pasamos la cadena binaria a su equivalente en *ascii*, también se implementa arboles binarios, tema visto en estructuras de datos.

Miguel Ángel: Cuando necesitamos comprimir información el algoritmo de Huffman es útil pues al generar códigos binarios para cada carácter que se repite, no es necesario utilizar todo el código ASCII, lo que representa un ahorro significativo en espacio si slo se utiliza cierto rango de caracteres. Este tipo de compresión de información no genera pérdida de información por lo cual se puede utilizar para cualquier tipo de información.

5 Bibliografía

References

[1] T. H, Cormen et al, *Introduction to Algorithms*. Ed third. Cambridge: The MIT Press, 2009.

[2] J. I *Análisis y diseño de algoritmos:* un enfoque teórico y práctico

[3] *A method for the construction of minimum-redundancy codes, Proceedings of the I.R.E., sept 1952, pp 10981102*

[4] *https://ronnyml.wordpress.com/2009/07/19/quicksort-en-c/*