Universitat de les Illes Balears

15-5-2021

ESTRUCTURA DE DATOS

Práctica II

Sergi Moreno Pérez

GRUPO 1 – 45185320A

DESCRIPCIÓN

INTRODUCCIÓN

Para realizar esta segunda práctica de la asignatura, debíamos implementar el algoritmo de codificación de *Huffman* mediante el uso de estructuras de datos vistas en clase:

- *Mapping* implementado con array indexado con claves
- Árbol binario
- *Heap*

TABLA DE FRECUENCIAS

Partiendo de un fichero de texto, debíamos generar una tabla de frecuencias con los caracteres presentes en el texto introducido en el fichero junto con su frecuencia de aparición en dicho texto. En su implementación, usamos un *mapping* ya que se trata de un conjunto de elementos caracteres que tienen asociado un único valor en otro conjunto de frecuencias.

Dentro de las diferentes opciones para implementar el *mapping* nos decidiremos por la estructura array indexado por claves porque estamos tratando un conjunto de elementos discreto de los que es probable que un gran número de ellos estén presentes en el texto. Así, mediante el uso de un iterador podremos verificar la existencia de los caracteres en el texto estableciendo su posición indexada en el array a True e iremos aumentando la frecuencia de aparición de estos caracteres existentes.

ÁRBOL DE HUFFMAN

ÁRBOLES DE UN SOLO NODO

Una vez tenemos creada la tabla de frecuencias, el siguiente paso será la generación de un árbol binario para cada carácter, donde tendremos un único nodo compuesto por el clave carácter y por el valor frecuencia de aparición. En nuestro programa tendremos que utilizar punteros a estos árboles debido a que los tipo árbol son TAD (Tipos Abstractos de Datos), los que están definidos como *limited*, impidiendo las operaciones de asignación y comparación de igualdad/desigualdad.

COLA DE PRIORIDADES (HEAP)

Ahora ya tendremos un árbol mononodo por cada carácter. Utilizaremos una cola de prioridades cuyos nodos serán punteros a dichos árboles para poder tener ordenados los diferentes caracteres en función a su aparición en el texto. Tendremos que definir si la prioridad será a mayor o a menor frecuencia, siendo para nosotros la segunda opción la adecuada ya que cuando queramos extraer los árboles de la cola para obtener el de *Huffman* necesitaremos que los más prioritarios sean los que tengan una frecuencia menor.

ÁRBOL DE HUFFMAN

Ya con todos los árboles ordenados en el *heap*, podremos ir obteniendo los dos más prioritarios en cada iteración para combinarlos y volverlos a introducir en el *heap*, consiguiendo así ir generando poco a poco el conocido como Árbol de *Huffman*.

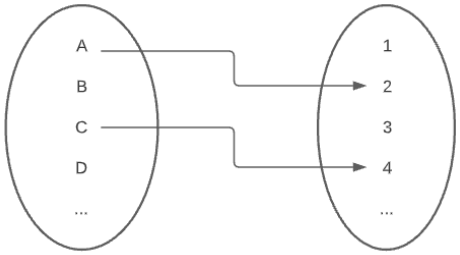
CODIFICACIÓN

Finalmente, solo restará realizar una búsqueda en preorden de cada clave carácter del conjunto, previamente habiendo comprobado si está presente en el conjunto o no. Cuando se haya verificado que la clave existe en el conjunto y se haya ubicado dentro del Árbol de *Huffman* se le asignará una codificación.

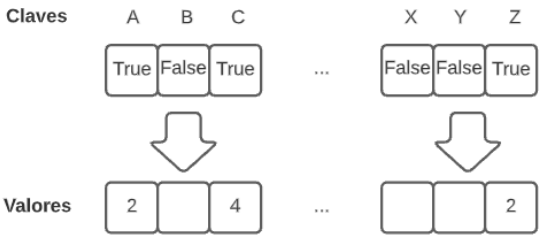
En el momento en el que se haya generado un código para cada carácter presente en el conjunto, podremos grabar en otro fichero externo el resultado de este algoritmo, encontrando cada clave con su código correspondiente.

DIAGRAMA DE LAS ESTRUCTURAS DE DATOS

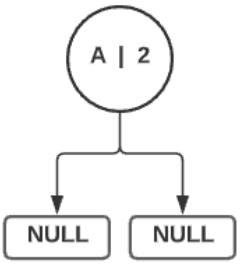
Estructura mapping



Mapping implementado con array indexado por claves

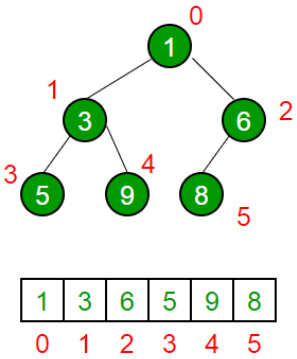


Árbol binario



Representación en array del Heap con nodos punteros

Representación general



Representación particular