Análisis de algoritmo de Huffman

Juan José Pizo Camacho

30000091781

jjpizoc@correo.usbcali.edu.co

Materia: Análisis de algoritmos

UNIVERSIDAD DE
SAN BUENAVENTURA

Universidad de San Buenaventura

Facultad de Ingeniería

Ingeniería de Sistemas

Cali

19 de noviembre del 2024

Huffman(C)		**Valores**
1	n = |C|	1
2	Q = C	$O(n)$ (con heapify)
3	**for** i = 1 to n - 1	$n-1$
4	allocate a new node z	1
5	z.left = x = EXTRACT-MIN(Q)	$2 * O(logn)$
6	z.right = y = EXTRACT-MIN(Q)	1
7	z.freq = x.freq + y.freq	1
8	INSERT(Q, z)	$O(log\ n)$
9	**return** EXTRACT-MIN(Q)	$O(log\ n)$

Total

Las operaciones en el bucle (líneas 4 a 8) se ejecutan n−1n - 1n−1 veces. El costo dominante dentro de cada iteración es: $O(logn) + O(logn) + O(logn) = O(logn)$

Por lo tanto, el costo del bucle es: $(n-1) \cdot O(logn) = O(nlogn)$

Los costos fuera del bucle son: $O(1) + O(n) + O(logn) = O(n)$

C**omplejidad total:**

$$O(nlogn)$$

Operaciones:

heapify: Inicializa la cola de prioridad en $O(n)$, convirtiendo el arreglo en un heap

EXTRACT-MIN **:** Cada extracción de un nodo del heap tiene un costo de $O(logn)$, ya que reorganiza el heap para mantener su propiedad

INSERT **:** Inserta un nuevo nodo en el heap en $O(logn)$

Costos dentro del bucle:

En este caso cada iteración del bucle realiza:

- Dos extracciones de mínimo $(2 \cdot O(logn))$.
- Un cálculo y creación de un nodo $(O(1))$.
- Una inserción en el heap $O(logn))$.

Total, por iteración: $\mathbf{3} \cdot \boldsymbol{O}(\boldsymbol{logn})$.

Dado que el bucle se ejecuta $n - 1$ veces, el costo total del bucle es:

$$Tbucle = (n - 1) \cdot O(logn) = O(nlogn)$$

Inicialización ($O(n)$) y la última extracción ($\boldsymbol{O}(\boldsymbol{logn})$).

Complejidad total:

$$\boldsymbol{T(n) = O(n) + O(nlogn) + O(logn) = O(nlogn)}$$

Memoria requerida

- **Heap:** El tamaño máximo del heap es n elementos.
- **Árbol de Huffman:** Se crean n−1 nodos adicionales para construir el árbol. Cada nodo tiene punteros a sus hijos y un valor de frecuencia.
- **Espacio adicional:** Variables auxiliares para almacenar referencias de nodos y frecuencias.

Complejidad de espacio: $\boldsymbol{O(n)}$