# *Árboles Binarios Ordenados*

Año 2024

## Introducción

Una de las estructuras de almacenamiento que se va a considerar en el ámbito de la materia es el *árbol binario*. Sin embargo, aún no se lo ha definido formalmente. Por su naturaleza, la definición más sencilla suele hacerse de manera recursiva. Este apunte estará dedicado a definirlo, dar algunas de las características o magnitudes que es posible analizar en ellos y también dar la definición de una de sus variantes. Además sobre el *árbol binario de búsqueda* se analizarán los costos de la localización tanto exitosa como la que fracasa, y como en esta estructura es necesario diferenciar entre el análisis a posteriori y el a priori, se van a evaluar ambos.

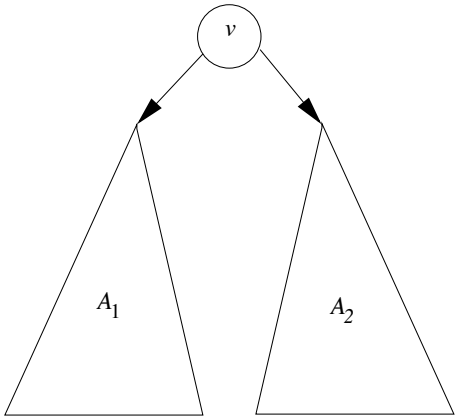
Para definir los árboles binarios que se van a usar como estructura de almacenamiento de relaciones, previamente es necesario considerar algunas definiciones:

**Definición:** Sea $\mathcal{A}_\mathcal{V}$ la familia de árboles binarios ordenados con valores en un conjunto $\mathcal{V}$, para lo que sigue definiremos que un árbol binario ordenado $A$ con valores en un conjunto $\mathcal{V}$ ($A \in \mathcal{A}_\mathcal{V}$), es decir un árbol binario en el que los nodos contienen valores de $\mathcal{V}$, recursivamente como:

1. $\Lambda \in \mathcal{A}_\mathcal{V}$
2. Si ($A_1 \in \mathcal{A}_\mathcal{V}$ y $A_2 \in \mathcal{A}_\mathcal{V}$ y $v \in \mathcal{V}$) entonces $A = \langle A_1, v, A_2 \rangle \in \mathcal{A}_\mathcal{V}$

$\diamond$

El calificativo de "binarios" en estos árboles se debe a que un árbol se define en general como un valor y *dos* subárboles, y el de "ordenados" es porque esos dos subárboles tienen una posición determinada en el árbol, lo cual se muestra en la siguiente gráfica:




Se obtendría por lo tanto un árbol distinto si $A_1$ apareciera a la derecha y $A_2$ a la izquierda.

En adelante se hablará simplemente de *árboles binarios* al referirse a árboles binarios ordenados.

Sobre los árboles es posible definir recursivamente algunas funciones tales como la altura $h$. En el caso de un árbol binario se define del siguiente modo.

**Definición:** Sea $\mathcal{A}_\mathcal{V}$ la familia de árboles binarios ordenados con valores en un conjunto $\mathcal{V}$, se define la *altura* $h : \mathcal{A}_\mathcal{V} \longmapsto \mathbb{N} \cup \{0\}$ de un árbol binario como:

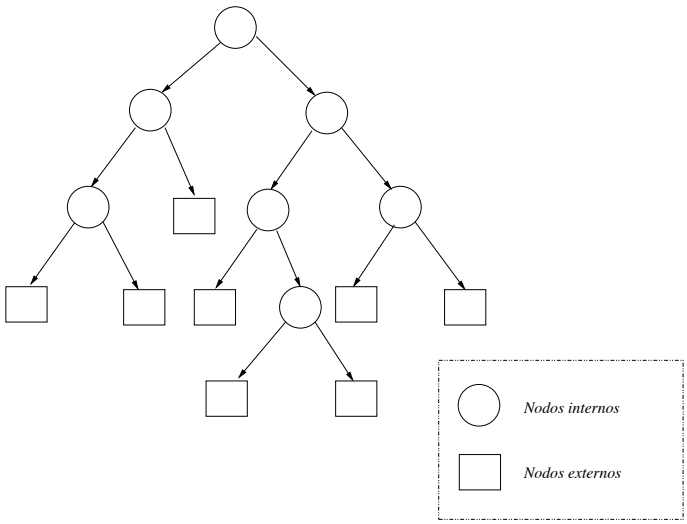
1. $h(\Lambda) = 0$
2. $h(\langle A_1, v, A_2 \rangle) = máx\{h(A_1), h(A_2)\} + 1$

◇

**Relación entre nodos internos y externos en un árbol binario**

Dado que se consideran los árboles binarios para almacenar elementos de un conjunto $\mathcal{V}$ (conjunto que se está representando en el árbol), se van a denominar nodos *internos* de un árbol a todos aquellos nodos que forman parte de él y que serán capaces de alojar elementos de $\mathcal{V}$. Mientras que los nodos *externos* van a representar los nodos que indican que ya no hay elementos; por lo tanto reemplazan los punteros a *nil* o las celdas que contienen marca de fin (dependiendo de cómo se indica que no hay más elementos).

La siguiente figura muestra un árbol binario en el que se distinguen los nodos externos y los internos:




Sean las siguientes definiciones recursivas de *cantidad de nodos externos* de un árbol $A$ ($e(A)$) y de *cantidad de nodos internos* de un árbol $A$ ($i(A)$).

**Definición:** Sea $\mathcal{A}_\mathcal{V}$ la familia de árboles binarios ordenados con valores en un conjunto $\mathcal{V}$, se define la *cantidad de nodos externos* $e : \mathcal{A}_\mathcal{V} \longmapsto \mathbb{N}$ de un árbol binario como:

1. $e(\Lambda) = 1$
2. $e(\langle A_1, v, A_2 \rangle) = e(A_1) + e(A_2)$

◇

**Definición:** Sea $\mathcal{A}_\mathcal{V}$ la familia de árboles binarios ordenados con valores en un conjunto $\mathcal{V}$, se define la *cantidad de nodos internos* $i : \mathcal{A}_\mathcal{V} \longmapsto \mathbb{N} \cup \{0\}$ de un árbol binario como:

1. $i(\Lambda) = 0$
2. $i(\langle A_1, v, A_2\rangle) = i(A_1) + i(A_2) + 1$

En un árbol binario $A$ se cumple la siguiente relación entre cantidad de nodos externos y nodos internos:

$$e(A) = i(A) + 1$$

**Demostración**

La demostración se puede realizar por inducción sobre el número de nodos internos del árbol.

*Paso Base:* Si el árbol es vacío entonces se tiene que:

$$e(A) = e(\Lambda) = i(\Lambda) + 1 = 0 + 1 = 1$$

*Hipótesis de inducción:* se asume que se cumple para todo árbol binario con cantidad de nodos internos menor que $n$.

*Se prueba entonces para cantidad de nodos internos igual a $n$:* Sea $A = \langle A_1, v, A_2\rangle$ con $n$ nodos internos.

$$\begin{aligned} e(A) &= e(A_1) + e(A_2) && (1)\\ &= i(A_1) + 1 + i(A_2) + 1 && (2)\\ &= i(A) + 1 && (3) \end{aligned}$$

en (1) se aplica la definición de $e$, en (2) se hace uso de la hipótesis de inducción, dado que la cantidad de nodos internos de $A_1$ y de $A_2$ es menor que $n$; en (3) se aplica la definición de $i$, y se llega finalmente a demostrar lo que buscaba (**c.q.d**).

□

**Relación entre suma de longitudes de paso a nodos internos y externos**

Sean las siguientes definiciones recursivas de *suma de longitudes de paso o camino a nodos externos* de un árbol $A$ ($E(A)$) y de *suma de longitudes de paso o caminos a nodos internos* de un árbol $A$ ($I(A)$).

**Definición:** Sea $\mathcal{A}_\mathcal{V}$ la familia de árboles binarios ordenados con valores en un conjunto $\mathcal{V}$, se define la *suma de longitudes de camino a nodos externos* $E : \mathcal{A}_\mathcal{V} \longmapsto \mathbb{N} \cup \{0\}$ de un árbol binario como:

1. $E(\Lambda) = 0$
2. $E(\langle A_1, v, A_2\rangle) = E(A_1) + E(A_2) + e(A)$

◇

**Definición:** Sea $\mathcal{A}_\mathcal{V}$ la familia de árboles binarios ordenados con valores en un conjunto $\mathcal{V}$, se define la *suma de longitudes de camino a nodos internos* $I : \mathcal{A}_\mathcal{V} \longmapsto \mathbb{N} \cup \{0\}$ de un árbol binario como:

1. $I(\Lambda) = 0$
2. $I(\langle A_1, v, A_2 \rangle) = I(A_1) + I(A_2) + i(A_1) + i(A_2)$

◇

En un árbol binario $A$ se cumple la siguiente relación entre suma de longitudes de paso a nodos externos y a nodos internos:

$$E(A) = I(A) + 2 \cdot i(A)$$

**Demostración**

La demostración se puede realizar también por inducción sobre el número de nodos internos del árbol.

*Paso Base:* Si el árbol es vacío entonces se tiene que:

$$\begin{aligned} E(A) &= E(\Lambda) \\ &= 0 \end{aligned} \tag{4}$$

$$\begin{aligned} I(A) + 2 \cdot i(A) &= I(\Lambda) + 2 \cdot i(\Lambda) \\ &= 0 + 2 \cdot 0 \\ &= 0 \end{aligned} \tag{5}$$

de (4) y (5) vemos que se cumple la igualdad.

*Hipótesis de inducción:* se asume que se cumple para todo árbol cuya cantidad de nodos internos es menor que $n$.

*Entonces, se prueba para cantidad de nodos internos igual a $n$:* Sea $A = \langle A_1, v, A_2 \rangle$ con $n$ nodos internos.

$$\begin{aligned} E(A) &= E(A_1) + E(A_2) + e(A) && (6) \\ &= (I(A_1) + 2 \cdot i(A_1)) + (I(A_2) + 2 \cdot i(A_2)) + e(A) && (7) \\ &= (I(A_1) + 2 \cdot i(A_1)) + (I(A_2) + 2 \cdot i(A_2)) + e(A_1) + e(A_2) && (8) \\ &= (I(A_1) + 2 \cdot i(A_1)) + (I(A_2) + 2 \cdot i(A_2)) + i(A_1) + 1 + i(A_2) + 1 && (9) \\ &= (I(A_1) + I(A_2) + i(A_1) + i(A_2)) + 2 \cdot i(A_1) + 2 \cdot i(A_2) + 1 + 1 && (10) \\ &= I(A) + 2 \cdot (i(A_1) + i(A_2) + 1) && (11) \\ &= I(A) + 2 \cdot i(A) && (12) \end{aligned}$$

en (6) se reemplaza $E(A)$ por su definición, en (7) se aplica la hipótesis de inducción sobre $E(A_1)$ y $E(A_2)$, dado que ambos subárboles tienen menos que $n$ nodos internos, en (8) se aplica la definición de $e(A)$, en (9) se usa la relación entre cantidad de nodos externos ($e$) e internos ($i$) sobre los subárboles $A_1$ y $A_2$, luego en (10) se asocia para armar la definición de $I(A)$ y en (11) se asocia y se saca factor común para obtener $i(A)$. Finalmente en (12) se obtiene la igualdad buscada (**c.q.d.**).

□

## Árbol Binario de Búsqueda

Hasta este momento, sólo se ha considerado un árbol binario como una representación de un conjunto, pero dicha representación no es útil aún para lograr eficiencia en las búsquedas.

Sea $\mathcal{A}_{\mathcal{V},\leq}$ la familia de los árboles binarios de búsqueda. Para definir el *árbol binario de búsqueda* $A$ sobre un conjunto de valores $\mathcal{V}$ ($A \in \mathcal{A}_{\mathcal{V},\leq}$) es necesario, además de contar con una relación de orden total sobre $\mathcal{V}$ ($\leq$) [1], definir una relación de orden sobre árboles y valores de $\mathcal{V}$ ($\preceq$) [2].

Se define recursivamente un árbol binario de búsqueda del siguiente modo.

**Definición:** La familia de los *árboles binarios de búsqueda* $\mathcal{A}_{\mathcal{V},\leq}$ sobre un conjunto de valores $\mathcal{V}$, siendo $\leq \subseteq \mathcal{V}^2$ una relación de orden total sobre $\mathcal{V}$, se define como:

1. $\Lambda \in \mathcal{A}_{\mathcal{V},\leq}$
2. Si ($A_1 \in \mathcal{A}_{\mathcal{V},\leq}$ y $A_2 \in \mathcal{A}_{\mathcal{V},\leq}$ y $v \in \mathcal{V}$ y $A_1 \preceq v$ y $v \prec A_2$) entonces $A = \langle A_1, v, A_2 \rangle \in \mathcal{A}_{\mathcal{V},\leq}$

$\diamond$

En esta definición si hubieran elementos iguales a $v$ en el árbol $A$ se deberían encontrar en el subárbol $A_1$ (lo mismo se puede tener por la recursividad en los subárboles $A_1$ y $A_2$). Una definición alternativa podría considerar que dichos elementos vayan al subárbol $A_2$ haciendo:

1. $\Lambda \in \mathcal{A}_{\mathcal{V},\leq}$
2. Si ($A_1 \in \mathcal{A}_{\mathcal{V},\leq}$ y $A_2 \in \mathcal{A}_{\mathcal{V},\leq}$ y $v \in \mathcal{V}$ y $A_1 \prec v$ y $v \preceq A_2$) entonces $A = \langle A_1, v, A_2 \rangle \in \mathcal{A}_{\mathcal{V},\leq}$

En nuestra materia se ha asumido que no se admiten elementos repetidos, pero el árbol binario de búsqueda no tiene inconvenientes para tratar con ellos y su definición lo evidencia (en la materia siguiente se estudiará cómo resolver la localización en este caso).

### Esfuerzos de localización a posteriori

Habiendo definido formalmente el árbol binario de búsqueda, se van a analizar ahora los esfuerzos medios y máximos de localización exitosa y localización que fracasa a posteriori sobre un árbol binario de búsqueda $A$ y para ello se usarán las funciones $h$, $E$ e $I$, definidas sobre árboles binarios.

Todos los nodos en el árbol son nodos internos y todos los punteros a nil son nodos externos (en otro caso las celdas con marca de fin serían los nodos externos). Se considerará como función de costo la *cantidad de celdas consultadas*. Una localización exitosa terminará por lo tanto en un nodo interno del árbol y la localización que fracasa en un nodo externo. De aquí que la longitud de paso a un nodo interno más uno nos dé el número de celdas consultadas para una localización exitosa y la longitud de paso a un nodo externo nos dé el número de celdas consultadas para una localización que fracasa.

[1] **Recordar** cuáles son las propiedades que debe tener la relación $\leq$ definida sobre $\mathcal{V}$ para que sea una relación de orden total (siendo $\leq \subseteq \mathcal{V} \times \mathcal{V} = \mathcal{V}^2$).

[2] La relación $\preceq$ se define entre árboles y valores de $\mathcal{V}$, y dice que: si $A \preceq v$ entonces todos los valores del árbol $A$ son menores o iguales que $v$ (bajo la relación $\leq$ definida en $\mathcal{V}$). **Queda como ejercicio definir formalmente la relación $\preceq$.**

***Esfuerzos máximos***

El esfuerzo máximo de de localización exitosa se da al buscar un $x$ ubicado en una celda que esté entre las de mayor nivel en el árbol, es decir una de las celdas que está más profunda. La cantidad de celdas que se consultan en ese caso coinciden con la *altura* del árbol. Por ejemplo, el árbol $A$ de la siguiente figura tiene altura $h(A) = 4$ y el esfuerzo máximo se daría al consultar por el elemento que se ubica en el nodo interno más profundo (en el nodo que está sombreado).

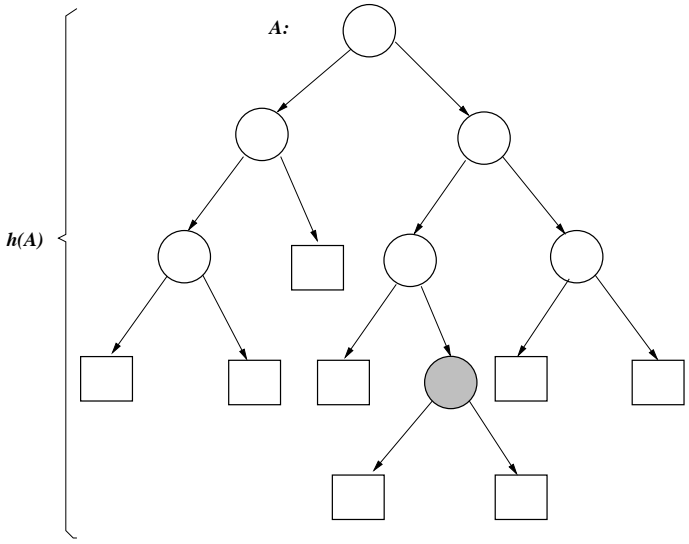



Figura 1: Ejemplo de árbol

Al analizar el esfuerzo máximo de localización que fracasa, se debe considerar que se está tratando de buscar un elemento $x$ que no está presente, al que se supone ubicado en un nodo externo; y que además, por ser esfuerzo máximo, debería ser alguno de los más profundos. Por consiguiente la cantidad de celdas consultadas en ese caso coincide nuevamente con la *altura* del árbol. [3] En el ejemplo del árbol de la Figura 1 el esfuerzo máximo de localización que fracasa se obtendría al intentar buscar un elemento menor, o también al buscar uno mayor, que el que contiene el nodo sombreado y entonces costaría $h(A) = 4$ consultas a celdas.

***Esfuerzos medios***

Se va a trabajar bajo *hipótesis de isoprobabilidad de consultar con éxito cualquiera de los $N$ elementos presentes* y suponiendo que *los $N + 1$ puntos de fracaso son también isoprobables*.

Se denotará con $C_N$ al esfuerzo medio de éxito en un árbol con $N$ elementos [4] y con $C'_N$ al esfuerzo medio de fracaso.

[3] Queda como ejercicio analizar si esto vale cuando los nodos externos representan a celdas conteniendo marca de fin en lugar de punteros a nil.

[4] Se usa $N$ en lugar de $i(A)$ para hacer explícita la cantidad de elementos del árbol y para mayor claridad.

Las siguientes fórmulas muestran cuánto valen dichos esfuerzos:

$$C_N = \frac{I(A) + N}{N} = \frac{I(A)}{N} + 1 = \frac{I(A)}{i(A)} + 1 \tag{13}$$

$$C'_N = \frac{E(A)}{N+1} = \frac{E(A)}{e(A)} \tag{14}$$

**Esfuerzos de localización a priori**

Para analizar los esfuerzos a priori hay que considerar los casos que se pueden presentar al construir todos los árboles para todas las secuencias de entrada posibles.

***Esfuerzos máximos***

Considerando nuevamente como función de costo la *cantidad de celdas consultadas*, se debe pensar en cuál sería el peor costo posible sobre todos los posibles árboles. Como el esfuerzo máximo en cada uno de ellos tiene que ver con la altura del árbol, se debe considerar cuál podría ser el árbol con mayor altura que se podría armar con $N$ elementos. En este caso, se pueden obtener varios árboles con la peor altura posible $h(A) = N$. Por lo tanto, como el esfuerzo máximo de localización exitosa y que fracasa coincidían, éstos son de $h(A) = N$ celdas consultadas.

***Esfuerzos medios***

Se analizará el esfuerzo medio de localización exitosa en un árbol binario de búsqueda en el cual se han incorporado los elementos de acuerdo a un orden aleatorio.

Se asumirá que las $N!$ *posibles secuencias de entrada son igualmente probables* para construir el árbol (o lo que es lo mismo, suponer que la entrada es aleatoria) y que todos *los elementos son igualmente probables de ser buscados con éxito*, y se continuará considerando como función de costo *cantidad de celdas consultadas*.

Por simplicidad se denotará con $I_N$ a la suma de las longitudes de paso a nodos internos y $E_N$ la suma de las longitudes de paso a nodos externos (si el árbol tiene $N$ nodos internos). Así se puede reescribir la fórmula que relacionaba a las funciones $i$, $I$ y $E$, para un árbol $A$ con $N$ nodos internos, como:

$$E_N = I_N + 2 \cdot N \tag{15}$$

y a las fórmulas de los esfuerzos medios de localización exitosa y localización que fracasa como:

$$C_N = \frac{I_N}{N} + 1 \tag{16}$$

$$C'_N = \frac{E_N}{N+1} \tag{17}$$

Para deducir el esfuerzo de localización exitosa a priori se aprovechará el hecho que los árboles binarios de búsqueda no reestructuran los elementos luego de una alta, entonces es posible relacionar los esfuerzos de localización exitosa, alta y localización que fracasa. Esto es porque la cantidad de celdas consultadas para localizar exitosamente un elemento del árbol es exactamente uno más que la necesaria para la localización que fracasó, indicando que dicho elemento no estaba en el árbol [5].

Por lo tanto, se puede decir que el costo a priori de localización exitosa para el primer elemento incorporado es uno más que el esfuerzo medio de localización que fracasa en un árbol con 0 elementos presentes; que el costo a priori de localizar el segundo elemento incorporado es uno más que el esfuerzo medio de localización que fracasa con 1 elemento presente y así sucesivamente. Entonces, y dado que se trabaja bajo hipótesis de isoprobabilidad de consultar cualquiera de los $N$ elementos, se puede decir que:

$$\begin{aligned} C_N &= 1 + \frac{C'_0 + C'_1 + C'_2 + \cdots + C'_{N-1}}{N} \\ &= 1 + \frac{\sum_{i=0}^{N-1} C'_i}{N} \end{aligned} \tag{18}$$

Además, se sabía que:

$$I_N = E_N - 2 \cdot N \tag{19}$$

al despejar de (15) $I_N$, y usando (16), es posible escribir:

$$\begin{aligned} C_N &= \frac{I_N}{N} + 1 \\ &= \frac{E_N - 2 \cdot N}{N} + 1 \\ &= \frac{E_N}{N} - 2 + 1 \\ &= \frac{E_N}{N} - 1 \end{aligned} \tag{20}$$

pero en (20) se puede usar (17) para conseguir:

$$\begin{aligned} C_N &= \frac{E_N}{N} \cdot \frac{N+1}{N+1} - 1 \\ &= C'_N \cdot \frac{N+1}{N} - 1 \\ &= C'_N \cdot \left(1 + \frac{1}{N}\right) - 1 \end{aligned} \tag{21}$$

[5] En lo que sigue se asumirá que no se realizan bajas en el árbol, porque sino no es posible asegurar que se mantenga la relación existente entre esfuerzos de localización exitosa, altas y localización que fracasa.

y juntando (18) y (21) se obtiene:

$$C'_N \cdot \left(1 + \frac{1}{N}\right) - 1 = 1 + \frac{\sum_{i=0}^{N-1} C'_i}{N} \quad (22)$$

$$N \cdot \left(1 + \frac{1}{N}\right) \cdot C'_N - N = N + \sum_{i=0}^{N-1} C'_i \quad (23)$$

$$(N+1) \cdot C'_N - 2 \cdot N = \sum_{i=0}^{N-1} C'_i \quad (24)$$

$$(N+1) \cdot C'_N = \sum_{i=0}^{N-1} C'_i + 2 \cdot N \quad (25)$$

desde (22) se saca común denominador a $N$ (en la parte derecha de la igualdad) y luego se pasa a $N$ multiplicando al otro miembro, desde (23) se distribuye el producto por $N$ y luego se pasa el término $N$ del lado derecho al lado izquierdo restando y se asocian los $N$ y desde (24) se pasa el término $2 \cdot N$ sumando al lado derecho para llegar a (25). Finalmente se logra una fórmula recursiva de $C'_N$.

Para resolver la recurrencia planteada, se reemplaza $N$ por $N-1$ en (25):

$$N \cdot C'_{N-1} = \sum_{i=0}^{N-2} C'_i + 2 \cdot (N-1) \quad (26)$$

$$= \sum_{i=0}^{N-2} C'_i + 2 \cdot N - 2 \quad (27)$$

ahora sustrayendo (27) a (25), miembro a miembro, se obtiene:

$$(N+1) \cdot C'_N - N \cdot C'_{N-1} = \sum_{i=0}^{N-1} C'_i + 2 \cdot N - \sum_{i=0}^{N-2} C'_i - 2 \cdot N + 2 \quad (28)$$

$$(N+1) \cdot C'_N - N \cdot C'_{N-1} = C'_{N-1} + 2 \quad (29)$$

$$(N+1) \cdot C'_N = (N+1) \cdot C'_{N-1} + 2 \quad (30)$$

$$C'_N = C'_{N-1} + \frac{2}{(N+1)} \quad (31)$$

desde (28) se simplifica lo más posible la fórmula, desde (29) se agrupan a la derecha los términos en que aparece $C'_{N-1}$, se los asocia y se saca a $C'_{N-1}$ como factor común, desde (30) se pasa dividiendo al otro término a $(N+1)$ para despejar finalmente a $C'_N$, y se obtiene así la fórmula (31) que define recursivamente a $C'_N$.

Ahora, dado que $C'_0 = 0$, es posible intentar resolver la recurrencia sustituyendo $C'_{N-1}$ por su

definición, luego $C'_{N-2}$ por su definición, y así sucesivamente hasta llegar a $C'_0$:

$$\begin{aligned} C'_N &= C'_{N-1} + \frac{2}{(N+1)} \\ &= C'_{N-2} + \frac{2}{N} + \frac{2}{(N+1)} \\ &= C'_{N-3} + \frac{2}{(N-1)} + \frac{2}{N} + \frac{2}{(N+1)} \\ &\vdots \\ &= C'_0 + \frac{2}{2} + \frac{2}{3} + \cdots + \frac{2}{(N-1)} + \frac{2}{N} + \frac{2}{(N+1)} \end{aligned}$$

reescribiendo la fórmula se tiene que:

$$C'_N = C'_0 + 2 \cdot \sum_{i=2}^{N+1} \frac{1}{i} \tag{32}$$

$$= C'_0 + 2 \cdot \left( \left( \sum_{i=1}^{N+1} \frac{1}{i} \right) - 1 \right) \tag{33}$$

$$= C'_0 + 2 \cdot H_{N+1} - 2 \tag{34}$$

$$= 2 \cdot H_{N+1} - 2 \tag{35}$$

en (32) se saca factor común 2 y se abrevian las sumas en una sumatoria, en (33) se suma y se resta 1 en el lado derecho de la igualdad y se cambia así el límite inferior de la sumatoria a 1, en (34) como la sumatoria corresponde al número armónico $N+1$, se la reemplaza por su nombre ($H_{N+1}$) y finalmente se reemplaza $C'_0$ por su valor.

$$C_N = (2 \cdot H_{N+1} - 2) \cdot \left(1 + \frac{1}{N}\right) - 1 \tag{36}$$

$$= \left(2 \cdot H_N + \frac{2}{(N+1)} - 2\right) \cdot \frac{(N+1)}{N} - 1 \tag{37}$$

$$= 2 \cdot H_N \cdot \frac{(N+1)}{N} + \frac{2}{(N+1)} \cdot \frac{(N+1)}{N} - 2 \cdot \frac{(N+1)}{N} - 1 \tag{38}$$

Ahora se puede reemplazar $C'_N$ en la fórmula (21) obteniendo:

$$= 2 \cdot H_N \cdot \left(1 + \frac{1}{N}\right) + \frac{2}{N} - 2 \cdot \left(1 + \frac{1}{N}\right) - 1 \quad (39)$$

$$= 2 \cdot H_N \cdot \left(1 + \frac{1}{N}\right) + \frac{2}{N} - 2 - \frac{2}{N} - 1 \quad (40)$$

$$= 2 \cdot H_N \cdot \left(1 + \frac{1}{N}\right) - 2 - 1 \quad (41)$$

$$= 2 \cdot H_N \cdot \left(1 + \frac{1}{N}\right) - 3 \quad (42)$$

$$\simeq 2 \cdot \ln N \cdot \left(1 + \frac{1}{N}\right) - 3 \quad (43)$$

Finalmente se llega a una fórmula para el esfuerzo medio de localización exitosa a priori (como subproducto también se obtuvo el esfuerzo medio de localización que fracasa a priori).

Se puede observar que la fórmula (43) obtenida en este desarrollo es la misma que aquélla que se puede obtener utilizando tanto el desarrollo algebraico puro, como el obtenido utilizando un desarrollo mixto (algebraico junto con funciones generatrices) [6].

La fórmula (43) que aproxima a $C_N$ evidencia que el esfuerzo medio de localización exitosa a priori en un árbol binario de búsqueda [7], es $O(\log N)$.

## Árboles Binarios de Búsqueda Balanceados en Altura (Árboles AVL)

Sea $A \in \mathcal{AB}_{\mathcal{V},\leq}$ la familia de los árboles binarios de búsqueda balanceados en altura. Se define un *árbol binario de búsqueda balanceado en altura* o *árbol AVL* ($A \in \mathcal{AB}_{\mathcal{V},\leq}$) [8] recursivamente de la siguiente manera.

**Definición:** Sea $\mathcal{AB}_{\mathcal{V},\leq}$ la familia de los *árboles binarios de búsqueda balanceados en altura* sobre un conjunto de valores $\mathcal{V}$, siendo $\leq \subseteq \mathcal{V}^2$ una relación de orden total sobre $\mathcal{V}$ y $h$ la altura de un árbol binario, se define recursivamente como:

1. $\Lambda \in \mathcal{AB}_{\mathcal{V},\leq}$
2. Si ($A_1 \in \mathcal{AB}_{\mathcal{V},\leq}$ y $A_2 \in \mathcal{AB}_{\mathcal{V},\leq}$ y $v \in \mathcal{V}$ y $A_1 \prec v$ y $v \prec A_2$ y $\mid h(A_1) - h(A_2) \mid \leq 1$) entonces $A = \langle A_1, v, A_2 \rangle \in \mathcal{AB}_{\mathcal{V},\leq}$

$\diamond$

Como ya se verá más adelante, este árbol necesita frente a las operaciones que pueden modificar la estructura, tales como altas y bajas, realizar ajustes para mantener el balance en las alturas.

[6] Ver la fórmula (23) del apunte sobre esfuerzo medio de localización exitosa en árbol binario de búsqueda.

[7] Observar que el esfuerzo máximo para el mejor árbol posible es $\simeq \log N$.

[8] El nombre de dichos árboles proviene de las iniciales de los dos matemáticos rusos que describieron esta estructura en 1962: *G. M. Adel'son-Vel'skiĭ y E. M. Landis*.

Notar que en la definición la relación de orden entre árboles y valores de $\mathcal{V}$ excluye la igualdad; por lo tanto, a diferencia del árbol binario de búsqueda, el árbol AVL no admite elementos repetidos. La razón de ello es que si se admitieran valores iguales, por alguna de las ramas, al realizar una operación para mantener el balance se puede perder la condición de ser un árbol binario de búsqueda (*Ejercicio: mostrar un ejemplo en que esto ocurra*).

Claramente los árboles AVL al mantener el *balance en altura* logran principalmente un mejor esfuerzo máximo de localización respecto del árbol binario de búsqueda, y por ello también mejoran el esfuerzo medio; pero esto se consigue a costa de que las operaciones de alta y baja sean más costosas.

En estos árboles no es posible relacionar los esfuerzos de localización exitosa, alta y localización que fracasa, aún bajo la hipótesis de que no existan bajas, debido a que para mantener el balance de las alturas se reestructura el árbol, y por ello un elemento no ocupa siempre la "posición" en que se dio de alta.

## Reconocimientos

El presente apunte se realizó tomando como base apuntes de clases del *Ing. Hugo Ryckeboer*, en la Universidad Nacional de San Luis y el libro **The Art of Computer Programming: *Sorting and Searching* (Vol. III)** de *Donald E. Knuth* [9].

[9] The Art of Computer Programming: *Sorting and Searching* (Vol. III), de D. Knuth, $2^{da}$ Edición, Addison–Wesley Pearson Education, ISBN: 0-201-89685-0.