Algoritmos y Estructuras de Datos I

Facultad de Ciencias Exactas y Tecnología
Universidad Nacional de Tucumán
2023

UNIVERSIDAD NACIONAL DE TUCUMÁN
facet
FACULTAD DE CIENCIAS EXACTAS Y TECNOLOGÍA

Búsqueda(2)

Arboles de Búsqueda

Los abb y los AVL son estructuras de datos usadas para hacer búsqueda interna, esto es, en las cuales la cantidad de datos se puede almacenar en memoria principal. Con estos árboles de puede buscar, insertar y borrar entradas de una tabla de tamaño n en el mejor caso en tiempo de orden $O(\log_2 n)$.

Esto se puede mejorar si se puede extender el concepto de árbol binario de búsqueda a un árbol de búsqueda más general, en el cual cada nodo admite más claves.

El árbol mas general de búsqueda se llama ***Arbol de Busqueda De M-vias***

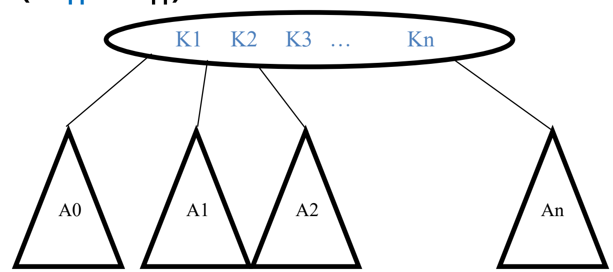
Un árbol de búsqueda de m-vías es un árbol en el que todos los nodos son de grado menor o igual que m, esto significa que cada nodo tiene a lo sumo m hijos.

Arbol De Búsqueda de M-Vias
(M-Way Search Tree)

Si T es un árbol vacío, entonces T es un árbol de búsqueda de m-vías.

Si T es no vacío, tiene las siguientes propiedades:

1) T tiene un nodo con: A_0, (K_1, A_1),(K_2, A_2),...,(K_n, A_n), $1 \leq n < m$
donde:
A_i, i=0,...,n son subárboles de T
K_i, i=1,...,n son valores de la clave



2) $K_i < K_{i+1}$ i=1,...,n-1

3) Todos los valores de las claves en el subárboles A_i son menores que K_{i+1} y mayores que K_i , i=1,...,n-1

4) Todos los valores de claves de A_n son mayores que K_n .

5) Todos los valores de claves de A_0 son menores que K_1 .

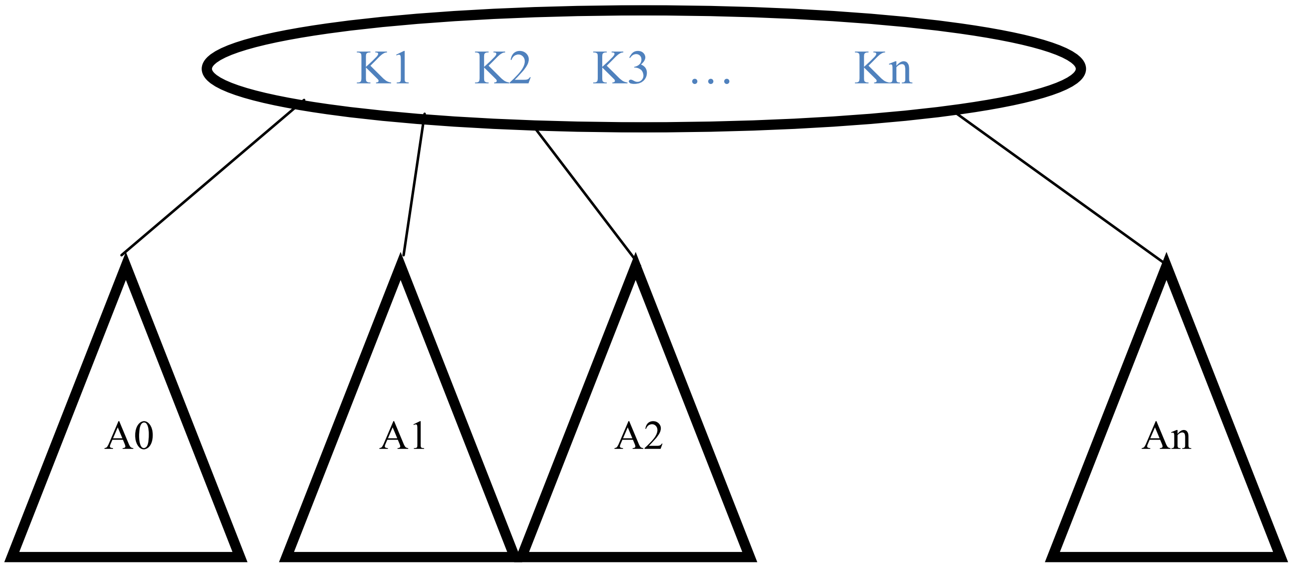
6) Los subárboles A_i, i=0,...,n son también arboles de búsqueda de m-vías.

Arbol De Búsqueda de M-Vias

Cada nodo de un árbol de búsqueda de m-vías tiene:

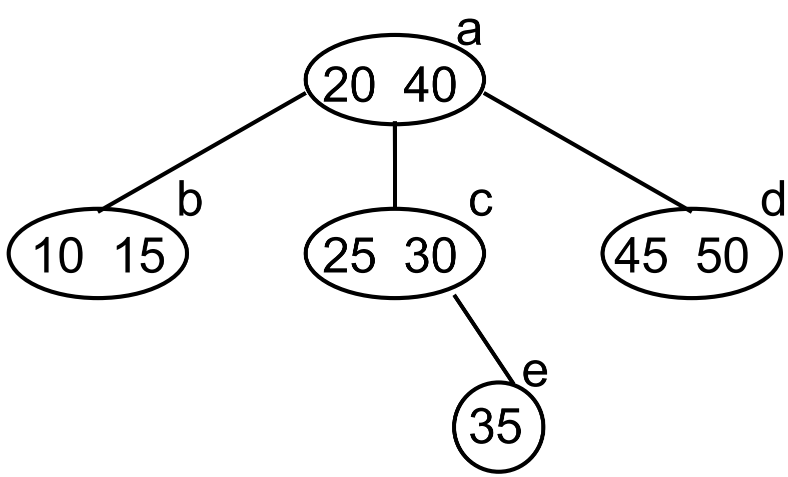
n claves: $K_1, K_2, ..., K_n$ $\quad 1 \leq n < m$

n+1 subarboles: $A_0, A_{1,}, A_2, A_n$ $\quad 1 \leq n < m$



Arbol De Búsqueda de M-Vias

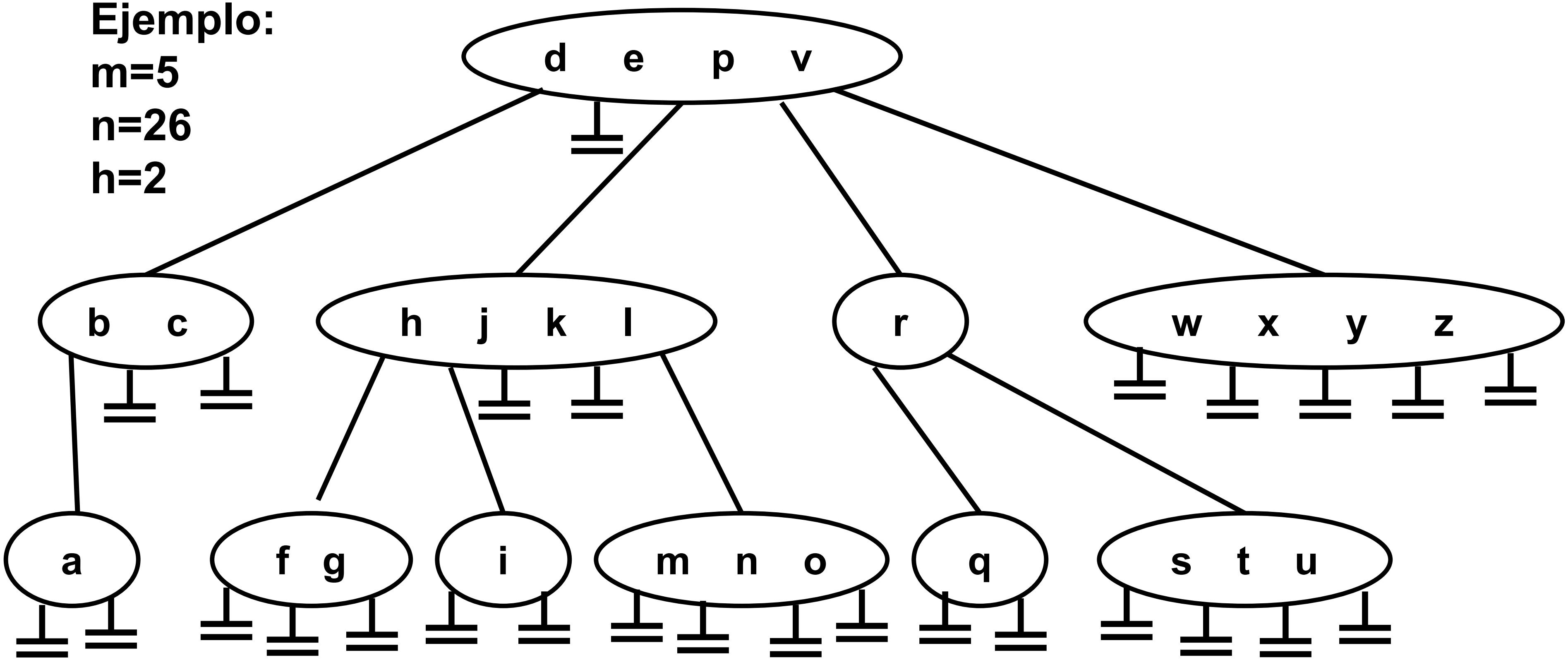
Ejemplo: árbol de búsqueda de 3 vías, 9 claves, altura h=2



a: b, (20,c), (40,d)
b: 0, (10,0), (15,0)
c: 0, (25,0), (30,e)
d: 0, (45,0), (50,0)
e: 0, (35,0)

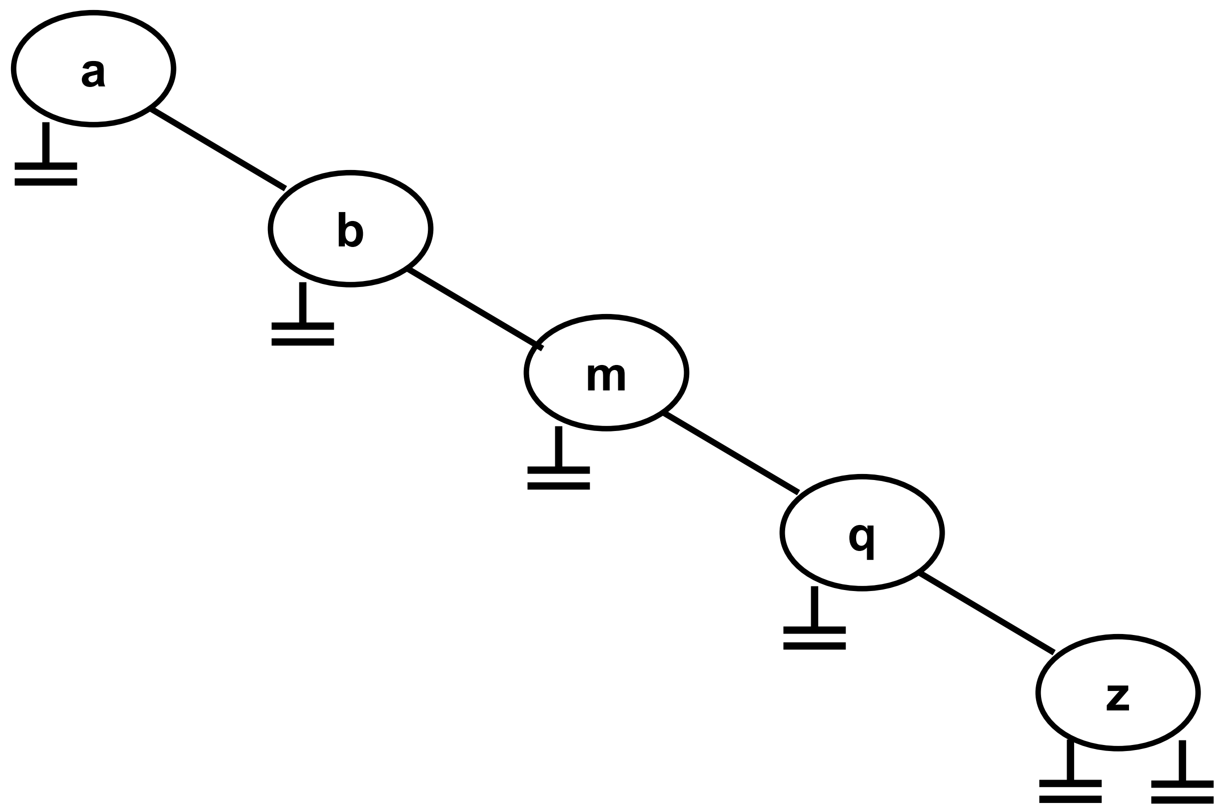
Arbol De Búsqueda de M-Vias

Ejemplo:
m=5
n=26
h=2



Arbol De Búsqueda de M-Vias

Ejemplo:
m=2
n=5
h=4



Arbol De Búsqueda de M-Vias

Numero de claves: **n**

Numero de vías: **m**

Altura: **h**

Cada nodo del árbol tiene como máximo: **m** subarboles y (**m-1**) claves.

Nodos:

$$Nodos_{max} = (m^{h+1} - 1)/(m-1) \qquad Nodos_{min} = h+1$$

Claves:

$$n_{max} = (m^{h+1} - 1) \qquad n_{min} = h+1$$

Altura:

$$h_{max} = n-1 \qquad h_{min} = \log_m(n+1) - 1$$

Arbol De Búsqueda de M-Vias

Por ejemplo:

- un árbol binario con h=3, admite n_{maximo}=7 claves,
- un árbol de 200 vías con h=3, admite n_{maximo}= $8*10^6$ -1 claves.

Para lograr el *mejor desempeño* de un árbol de m vías se necesita un **árbol de búsqueda de m vías óptimo**. Un árbol de este tipo es difícil de mantener ya que los algoritmos son poco eficientes.

Para lograr un *buen desempeño* se puede considerar un **árbol de búsqueda de m vías balanceado**. No es difícil mantener el balance en árboles de m vías. Los árboles de m vías casi balanceados llamados **árboles B** son más populares y útiles que los árboles de m vías totalmente balanceados.

ARBOL-B

Definición: Un **árbol-B de orden m** es un ***árbol de m vías*** ***balanceado*** tal que:

- T es un árbol vacío
- T es un árbol un árbol de búsqueda de m-vías, de altura $\geq$ 1 tal que satisface las siguientes propiedades:

1) La raíz tiene al menos dos hijos.
2) Todos los nodos excepto la raíz tienen al menos $\lceil m/2 \rceil$ hijos.
3) Todas las hojas del árbol están al mismo nivel.

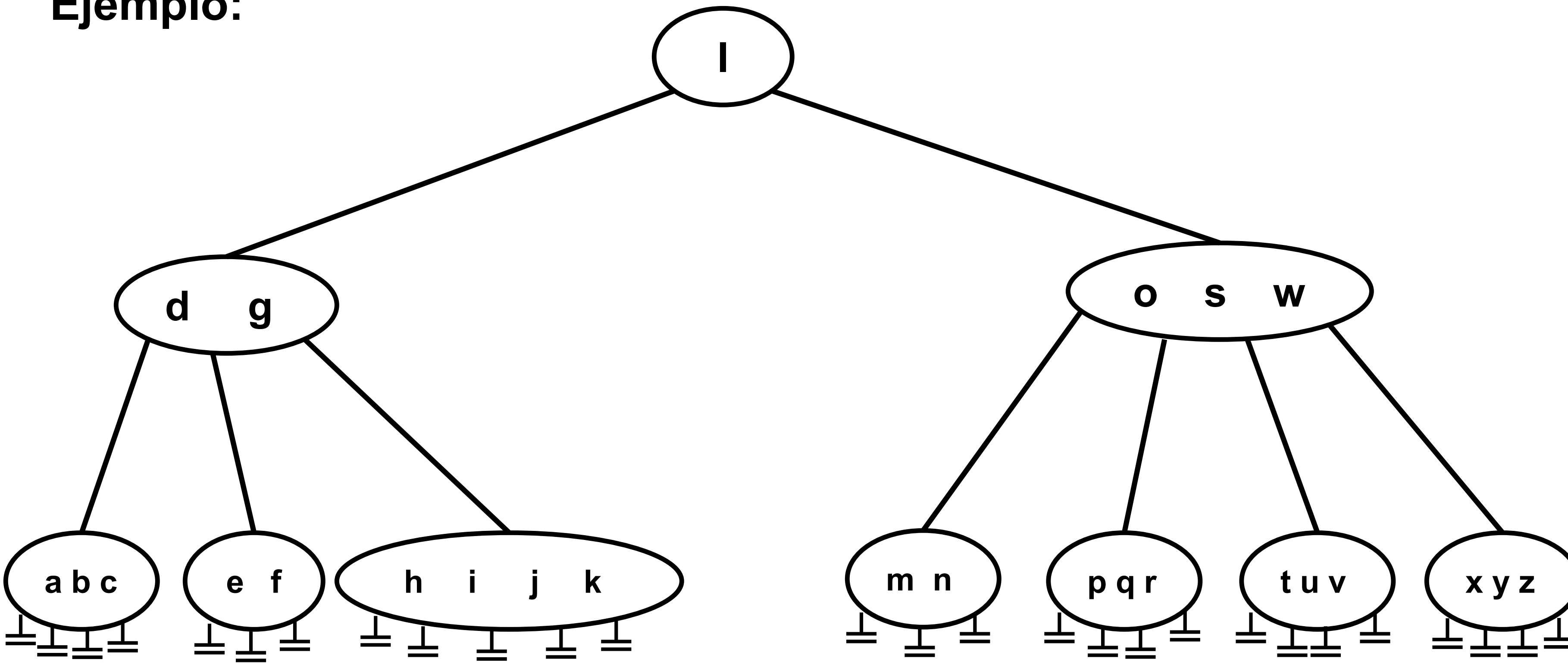
Altura:

$$\log_m(n+1)-1 \leq h \leq 1+\log_{\lceil m/2 \rceil}((n+1)/2), \quad m>2$$

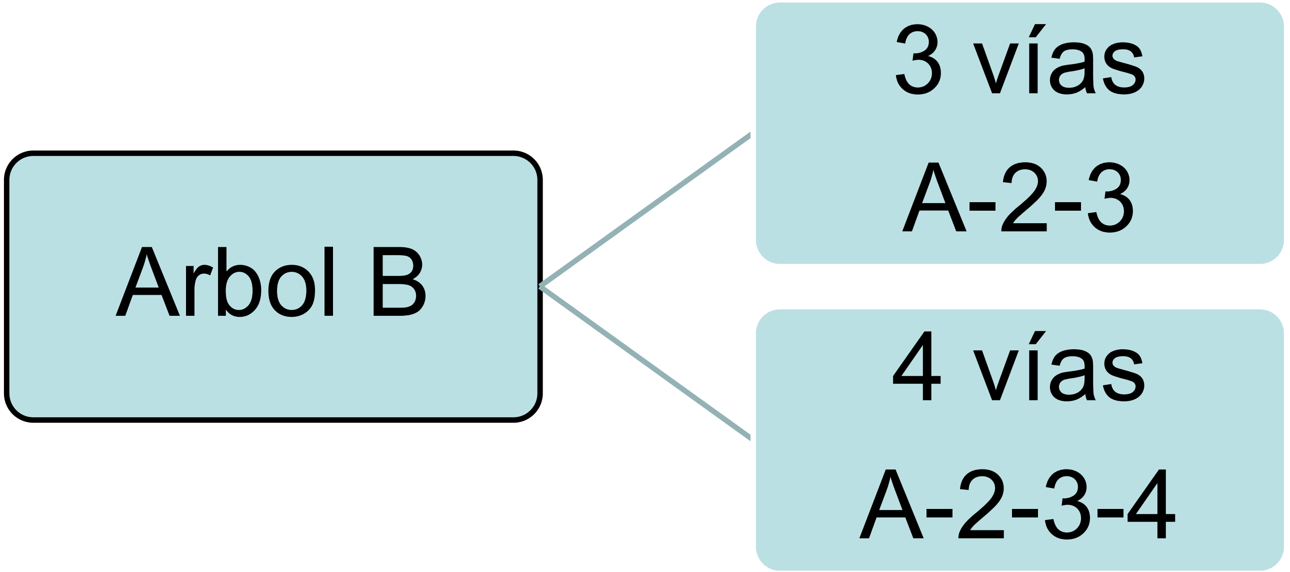
R. Bayer, E. McCreight ."*Organization and Maintenance of Large Ordered Indexes*".
Acta Informatica, Vol. 1, Fasc. 3, 1972 pp. 173-189

ARBOL-B

Ejemplo:



ARBOL-B



Árbol 2-3 de búsqueda

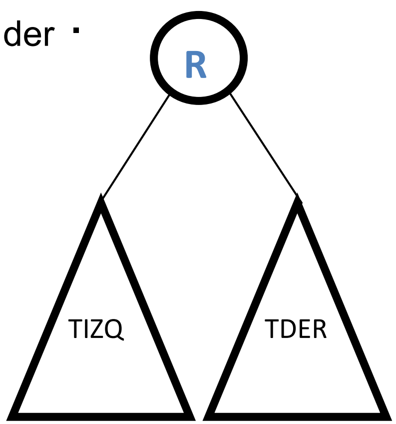
Un árbol 2-3 es un árbol tal que sus nodos interiores pueden tener grado 2 o 3 y sus hojas están todas al mismo nivel.

Definición: Un árbol T es un árbol 2-3 de altura h si:

- T es **vacío.** (h=0)

- Si T es un **nodo-2** con raíz con clave **R** y con T_{izq} y T_{der} como subárboles.

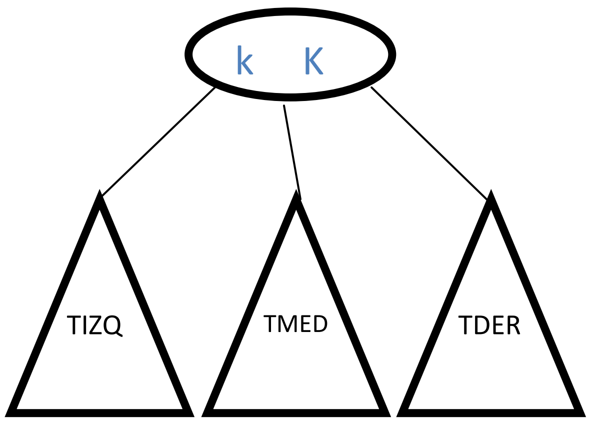
a) T_{izq} y T_{der} son árboles 2-3 de búsqueda de altura h-1.
b) clave de T_{izq} < **R** < claves de T_{der} .



Árbol 2-3 de búsqueda

- Si T es un **nodo-3** con raíz con claves **k** y **K** y con T_{izq} , T_{med} y T_{der} como subárboles.

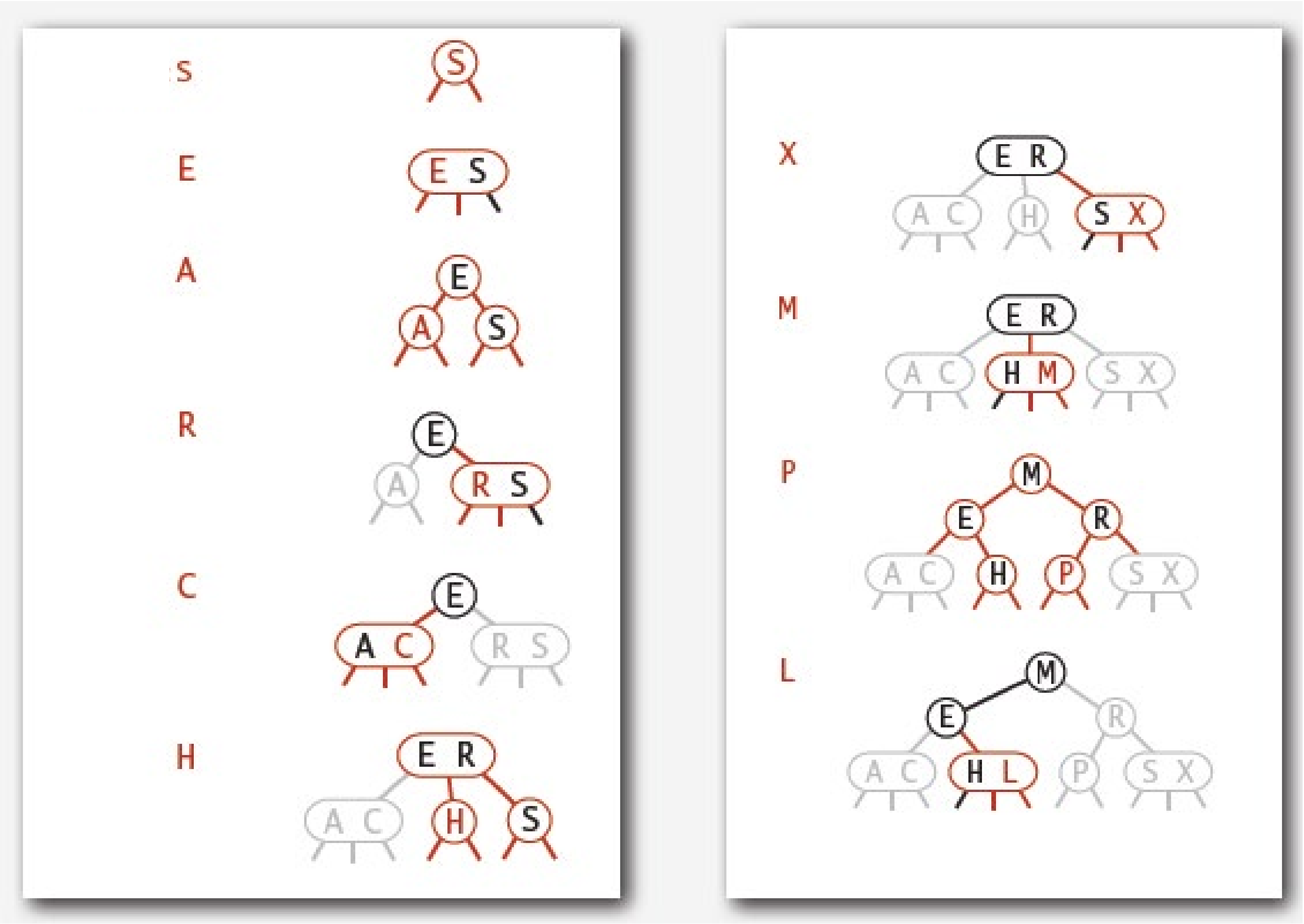
a) T_{izq} , T_{med} y T_{der} son árboles 2-3 de búsqueda de altura h-1.
b) claves de T_{izq} < **k** < claves de T_{med} .
c) claves de T_{med} < **K** < claves de T_{der} .



- Todas las claves son distintas entre si.

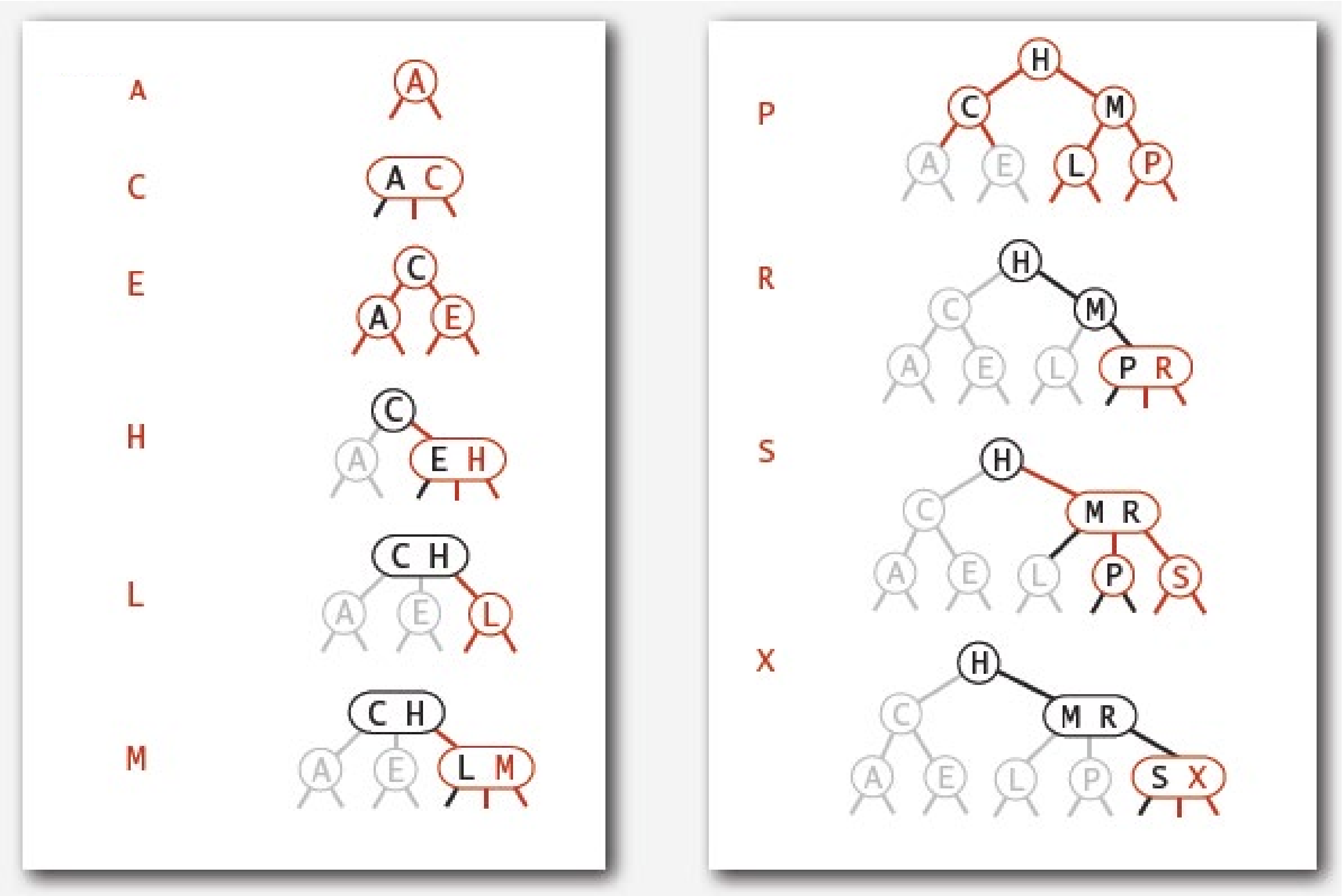
Árbol 2-3 de búsqueda

Ejemplo 1: Construcción del árbol: **SEARCHXMPL**



Árbol 2-3 de búsqueda

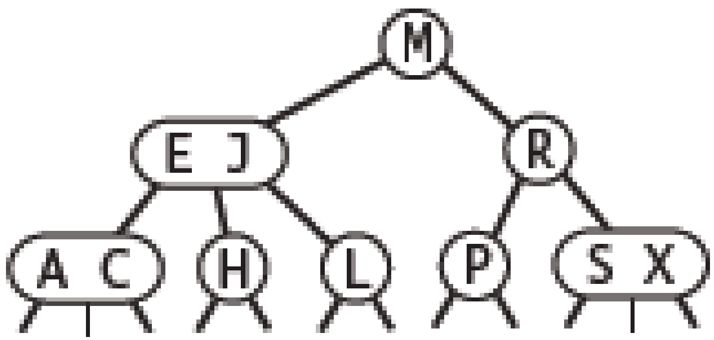
Ejemplo 2: Construcción del árbol: **ACEHLMPRST**



Árbol 2-3 de búsqueda

Ejemplo:

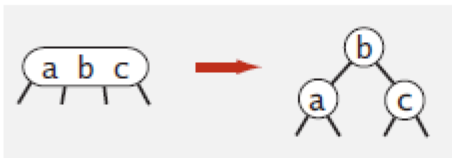
Búsqueda, Inserción, Construcción.



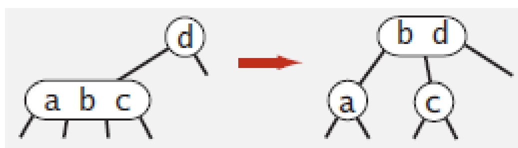
Árbol 2-3 de búsqueda

Transformación de nodos-4:

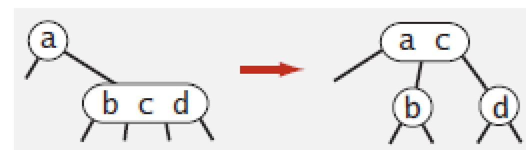
Raiz:



Padre nodo-2: izquierdo

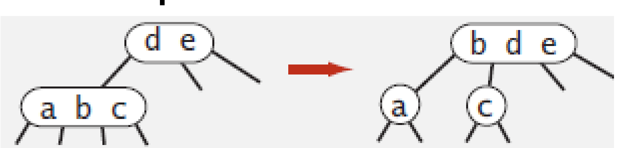


derecho

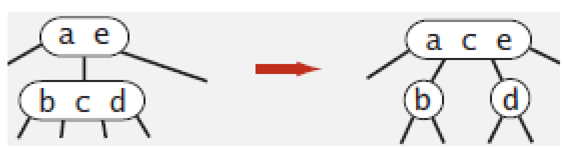


Padre nodo-3:

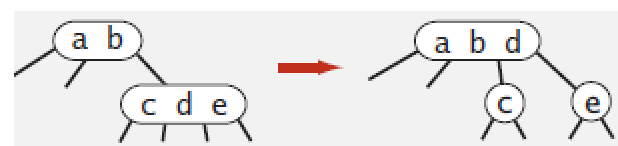
izquierdo



medio



derecho



Árbol 2-3 de búsqueda

Altura: Un árbol 2-3 con **n** claves tiene una altura h:

$$\lceil \log_3 (n+1) \rceil -1 \leq h \leq \lceil \log_2 (n+1) \rceil -1$$

Claves: Un árbol 2-3 de altura **h** tiene claves entre:

$$n_{min} = (2^{h+1} - 1) \qquad n_{max} = (3^{h+1} - 1)$$

Operaciones: en un árbol 2-3 se pueden realizar las operaciones de:

Buscar un nodo con una clave dada
Insertar un nodo con una clave dada
Borrar un nodo con una clave dada

peor caso con costo
T $\in O(\log_2 n)$

Arbol 2-3-4 de búsqueda

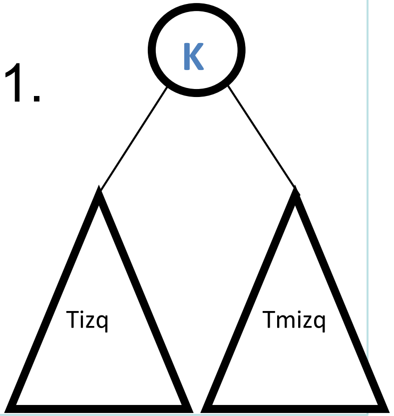
Un árbol 2-3-4 es un árbol tal que sus nodos interiores pueden tener grado 2 , 3 o 4 y sus hojas están todas al mismo nivel.

Definición: Un árbol T es un árbol 2-3-4 de altura h si:

- T es **vacío.** (h=0)

- Si T es un **nodo-2** con raíz con una sola clave **K** y con T_{izq} y T_{mizq}

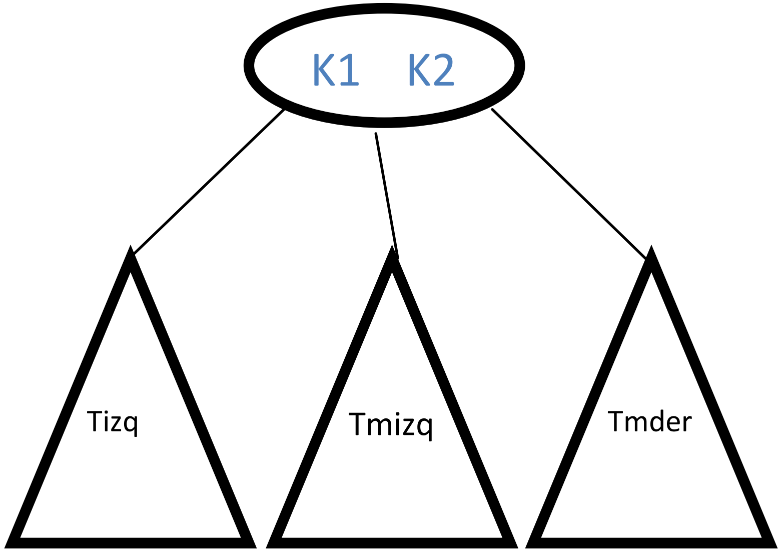
a) T_{izq} y T_{mizq} son árboles 2-3-4 de búsqueda de altura h-1.
b) clave de T_{izq} < **K** < claves de T_{mizq} .



Arbol 2-3-4 de búsqueda

- Si T es un **nodo-3** con raíz con claves **k1** y **k2** y con T_{izq} , T_{mizq} y T_{mder}

a) T_{izq} , T_{mizq} y T_{mder} son árboles 2-3-4 de búsqueda de altura h-1.
b) claves de T_{izq} < **k1** < claves de T_{mizq} < **k2** < claves de T_{mder} .

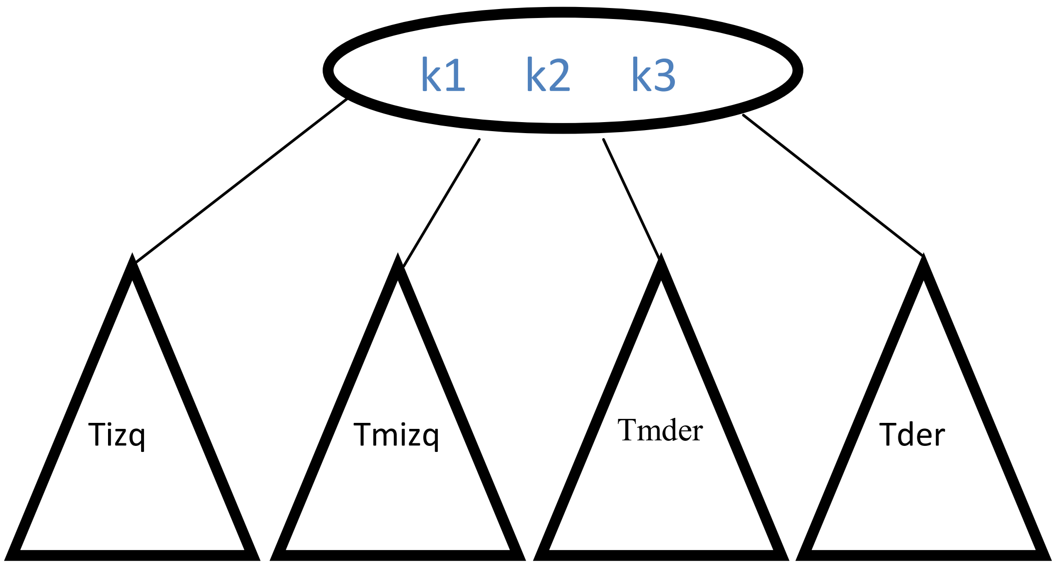


Arbol 2-3-4 de búsqueda

- Si T es un **nodo-4** con claves **k1, k2** y **k3** ,con T_{izq} , T_{mizq} , T_{mder} ,T_{der}

a)T_{izq} , T_{mizq} , T_{mder} y T_{der} son árboles 2-3-4 de búsqueda de altura h-1.

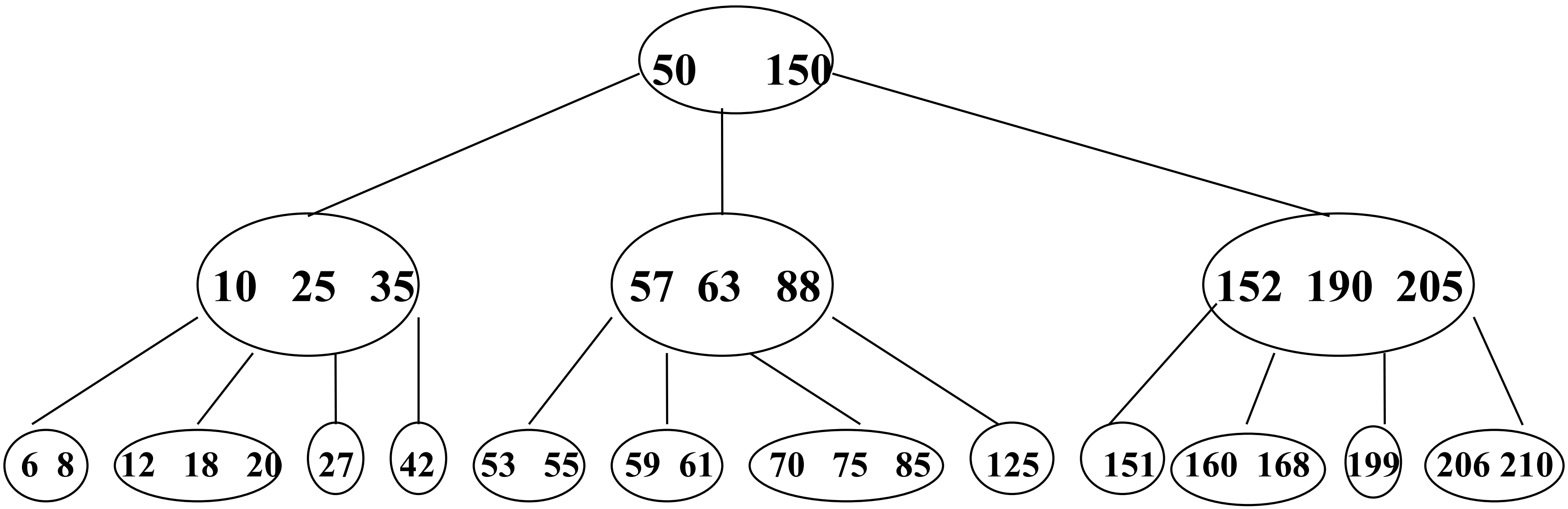
b) claves T_{izq} < **k1** < claves T_{mizq} < **k2** < claves T_{mder} < **k3** < claves T_{der}



- Todas las claves son distintas entre si.

Arbol 2-3-4 de búsqueda

Ejemplo:



Arbol 2-3-4 de búsqueda

Altura: Un árbol 2-3-4 con **n** claves tiene una altura h:

$$\lceil \log_4 (n+1) \rceil -1 \leq h \leq \lceil \log_2 (n+1) \rceil -1$$

De modo que: $h_{min} = \lceil \log_4 (n+1) \rceil -1$ $\quad h_{max} = \lceil \log_2 (n+1) \rceil -1$

Claves: Un árbol 2-3-4 de altura **h** tiene claves entre:

$$(2^{h+1} -1) \leq n \leq (4^{h+1} -1)$$

De modo que: $n_{min} = (2^{h+1} -1)$ $\quad n_{max} = (4^{h+1} -1)$

Operaciones: en un árbol 2-3-4 se puede realizar las operaciones de:

- **Buscar** un nodo con una clave dada
- **Insertar** un nodo con una clave dada
- **Borrar** un nodo con una clave dada

peor caso con costo T ϵ $O(\log_2 n)$

Arbol B

Los algoritmos de insertar un nodo y de borrar un nodo son más simples en el árbol 2-3-4 que en el árbol 2-3.
Un árbol 2-3-4 se puede representar como un árbol binario (llamado árbol rojo-negro), que resulta muy eficiente en la utilización del espacio.-

Existen muchas variantes de los Arboles B, como por ejemplo:

Arbol B+, Arbol B*, Arbol B#.

Los arboles B y sus variantes son ampliamente usados para sistemas de archivos y bases de datos:

- Windows: NTFS.
- Mac: HFS, HFS+.
- Linux: ReiserFS, XFS, Ext3FS, JFS.
- Databases: ORACLE, DB2, INGRES, SQL, PostgreSQL.