

Algoritmos y Estructuras de Datos

Cursada 2011

Prof. Catalina Mostaccio

Prof. Alejandra Schiavoni

Facultad de Informática - UNLP



Árboles Binarios de Búsqueda



Agenda

- ❖ Árbol Binario de Búsqueda
- ❖ Árboles AVL



Árbol Binario de Búsqueda: Definición

Un árbol binario de búsqueda es una colección de nodos conteniendo claves, que debe cumplir con una propiedad estructural y una de orden.



Árbol Binario de Búsqueda

La propiedad estructural: es un árbol binario

La propiedad de orden, es la siguiente:

*para cada nodo N del árbol se cumple que todos los nodos ubicados en el **subárbol izquierdo** contienen claves **menores** que la clave del nodo N y los nodos ubicados en el **subárbol derecho** contienen claves **mayores** que la clave del nodo N*



Árboles AVL

- ❖ Definición
- ❖ Características
- ❖ Inserción
- ❖ Desbalanceo
- ❖ Rotaciones Simples y Dobles
- ❖ Eliminación



Árbol AVL: Definición

Un árbol AVL (Adelson–Velskii–Landis) es un árbol binario de búsqueda que cumple con la condición de estar balanceado

- La propiedad de balanceo que cumple dice:

Para cada nodo del árbol, la diferencia de altura entre el subárbol izquierdo y el subárbol derecho es a lo sumo 1



Características

- *La propiedad de balanceo es fácil de mantener y garantiza que la altura del árbol es de $O(\log n)$*
- *En cada nodo del árbol se guarda información de la altura*
- *La altura del árbol vacío es -1*
- *Se debe mantener el balanceo al realizar las operaciones sobre el árbol (inserción y borrado)*



Operaciones en un AVL

➤ *Inserción*

➤ *Eliminación*



Inserción en el árbol AVL

- *La inserción se realiza igual que en un árbol binario de búsqueda*
- *Puede destruirse la propiedad de balanceo*



Rebalancear el árbol

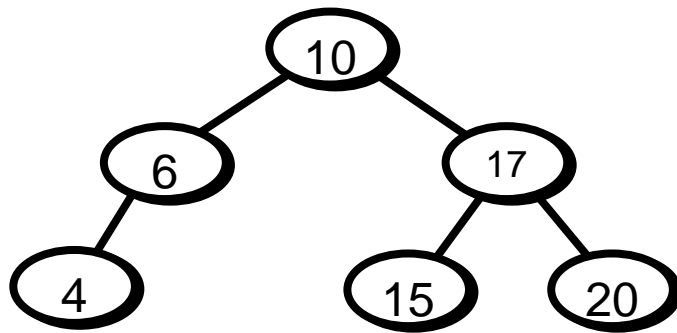


Problemas: Desbalanceo

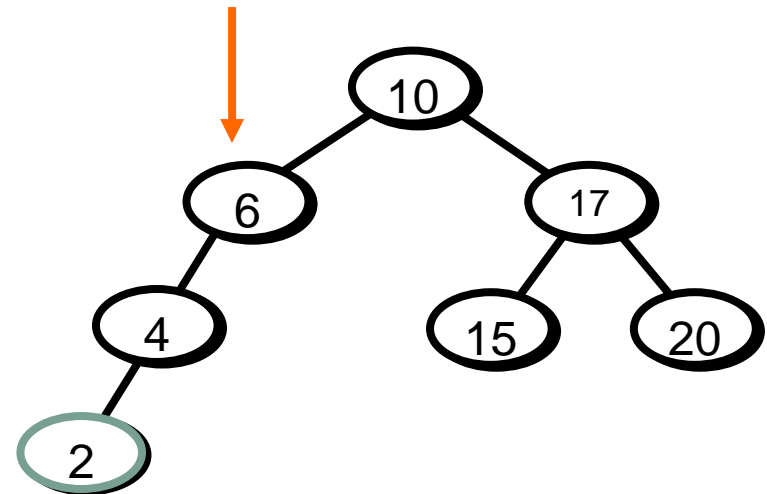
- *Al insertar un elemento se actualiza la información de la altura de los nodos que están en el camino desde el nodo insertado a la raíz*
- *El desbalanceo sólo se produce en ese camino, ya que sólo esos nodos tienen sus subárboles modificados*

Problemas: Desbalanceo

Ejemplo al insertar un nodo



Se desbalancea el 6



Árbol después de insertar el 2



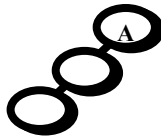
Rebalanceo del árbol

- *Para restaurar el balanceo del árbol:*
 - *se recorre el camino de búsqueda en orden inverso*
 - *se controla el equilibrio de cada nodo*
 - *Si está desbalanceado se realiza una modificación simple: rotación*
 - *después de rebalancear el nodo, la inserción termina*
 - *este proceso puede llegar a la raíz*

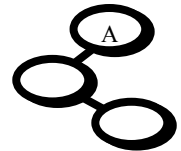
Rebalanceo del árbol

Hay 4 casos posibles de desbalanceo a tener en cuenta, según donde se hizo la Inserción. El nodo A es el nodo desbalanceado.

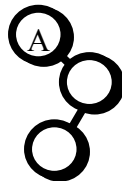
1. Inserción en el Subárbol IZQ del hijo IZQ de A



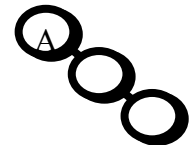
2. Inserción en el Subárbol DER del hijo IZQ de A



3. Inserción en el Subárbol IZQ del hijo DER de A



4. Inserción en el Subárbol DER del hijo DER de A





Rebalanceo del árbol

- *La solución para restaurar el balanceo es la **ROTACION***

*La **rotación** es una modificación simple de la estructura del árbol, que restaura la propiedad de balanceo, preservando el orden de los elementos*



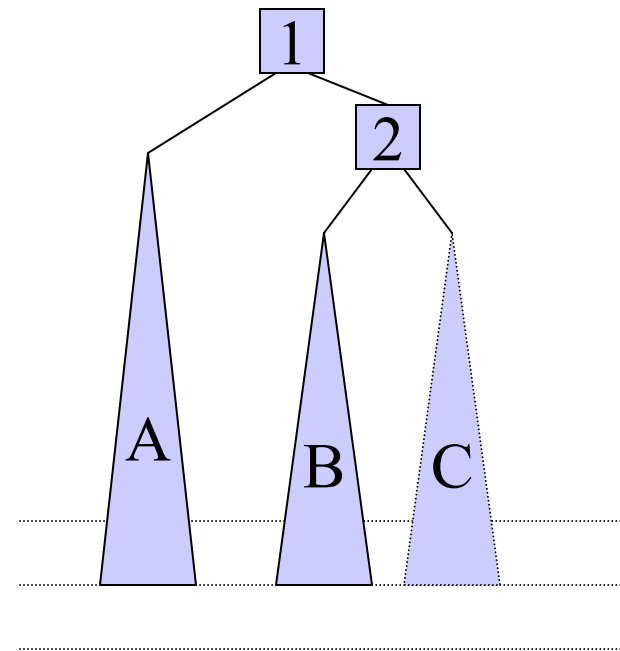
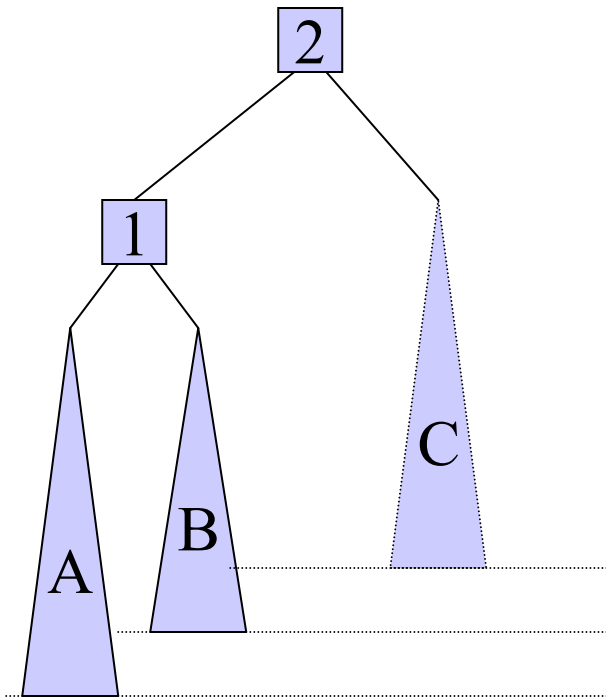
Rebalanceo del árbol

- *Existen dos clases de rotaciones:*
 - *Rotación Simple: Casos 1 y 4: inserción en el lado externo*
 - *Rotación Doble: Casos 2 y 3: inserción en el lado interno*
- *Soluciones simétricas: En cada caso, los subárboles están opuestos.*



Rotación Simple

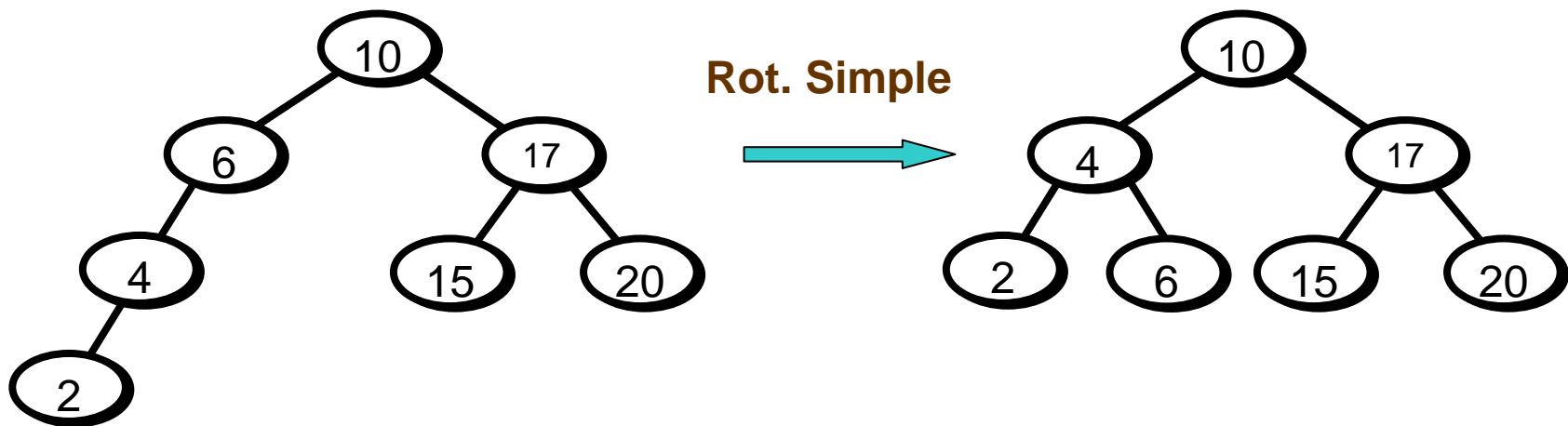
Caso 1: Rotación Simple Izq-Izq



Se obtuvo nuevamente un árbol balanceado

Rotación Simple (cont.)

Siguiendo con el ejemplo:





Rotación Simple (cont.)

Caso 4: Rotación Simple Der-Der

Es simétrico al caso 1, el desbalanceo se produce hacia el lado derecho

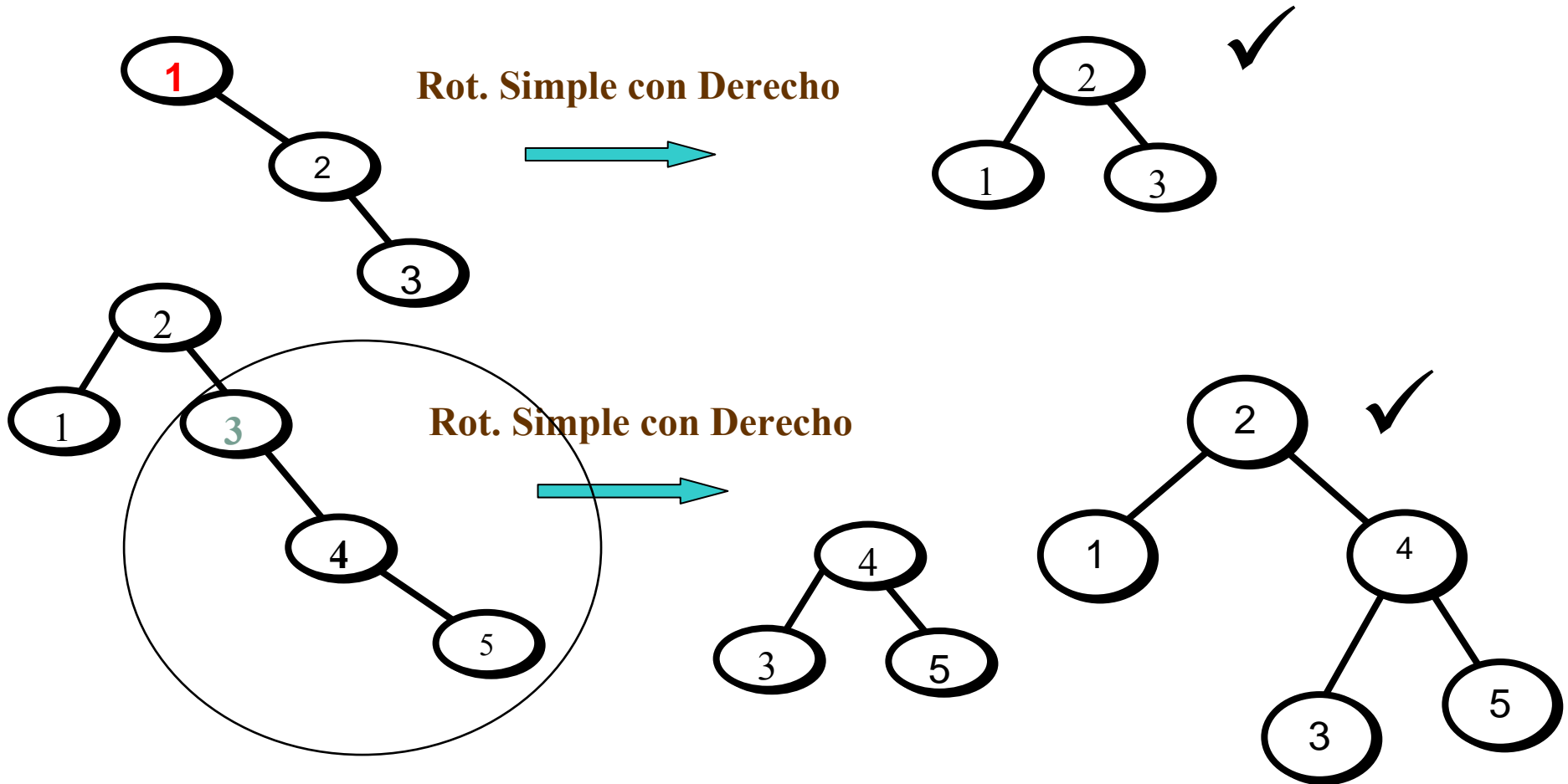


Rotación Simple (cont.)

Ejemplo:

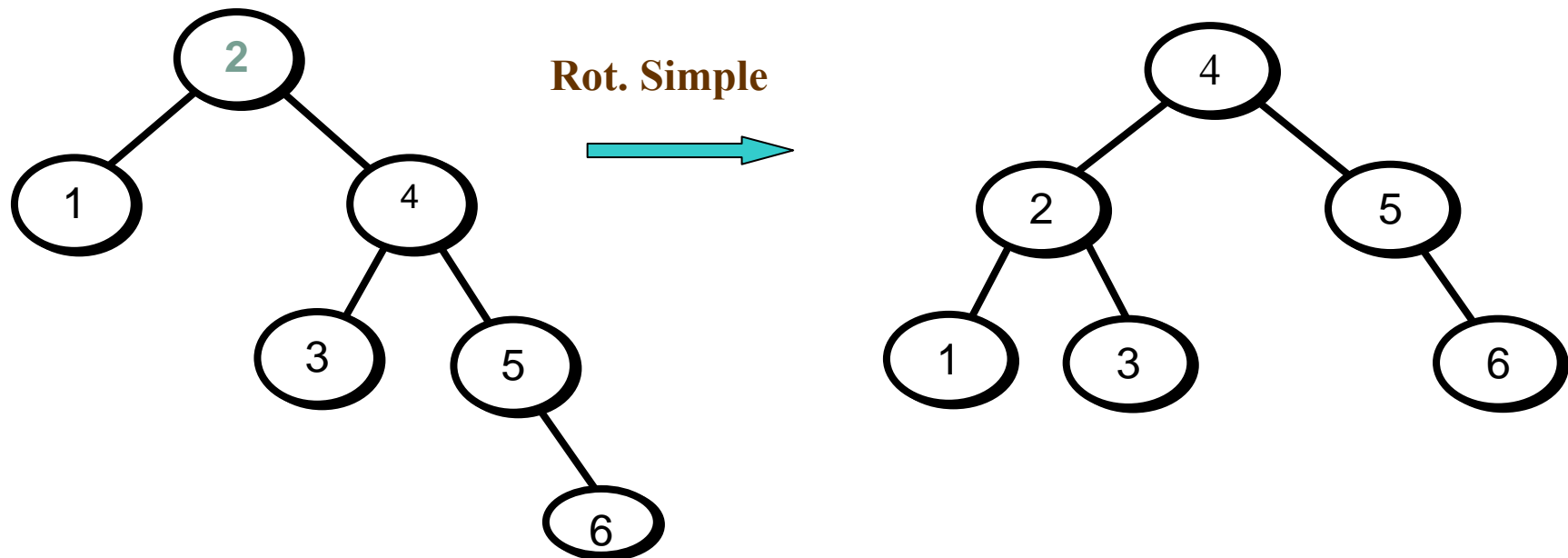
*Insertar las claves del 1 al 7 en ese orden,
en un árbol AVL inicialmente vacío*

Rotación Simple (cont.)



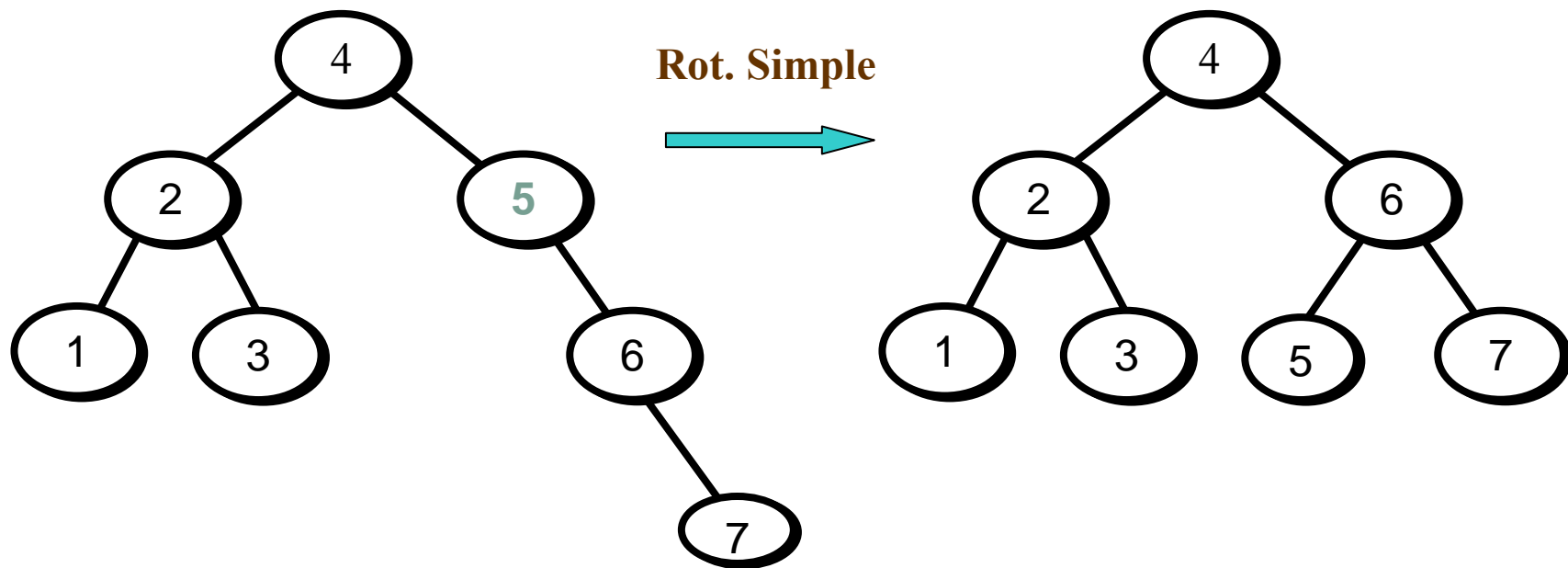
Rotación Simple (cont.)

Siguiendo con el ejemplo:



Rotación Simple (cont.)

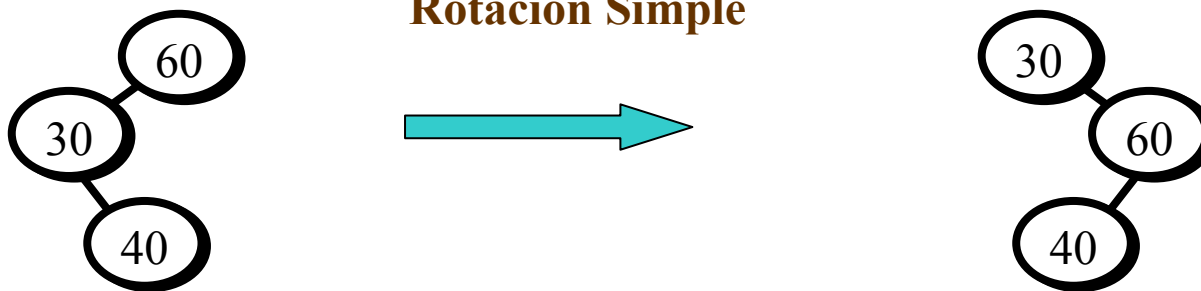
Siguiendo con el ejemplo:



Rotación Doble

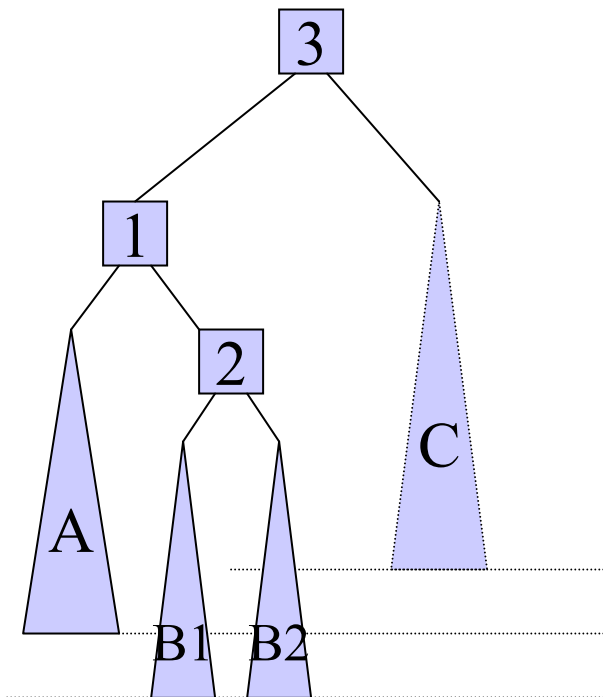
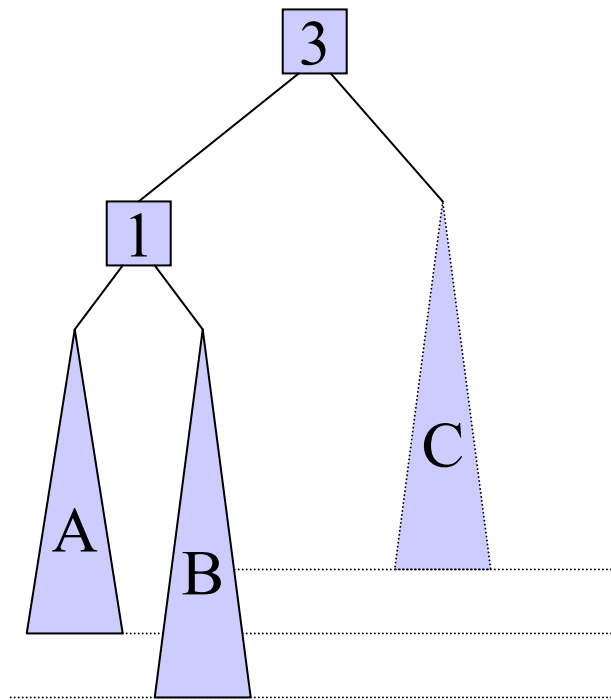
- *En algunos casos la rotación simple no resuelve el problema*

Caso 2: Rotación Doble Izq-Der



Rotación Doble (cont.)

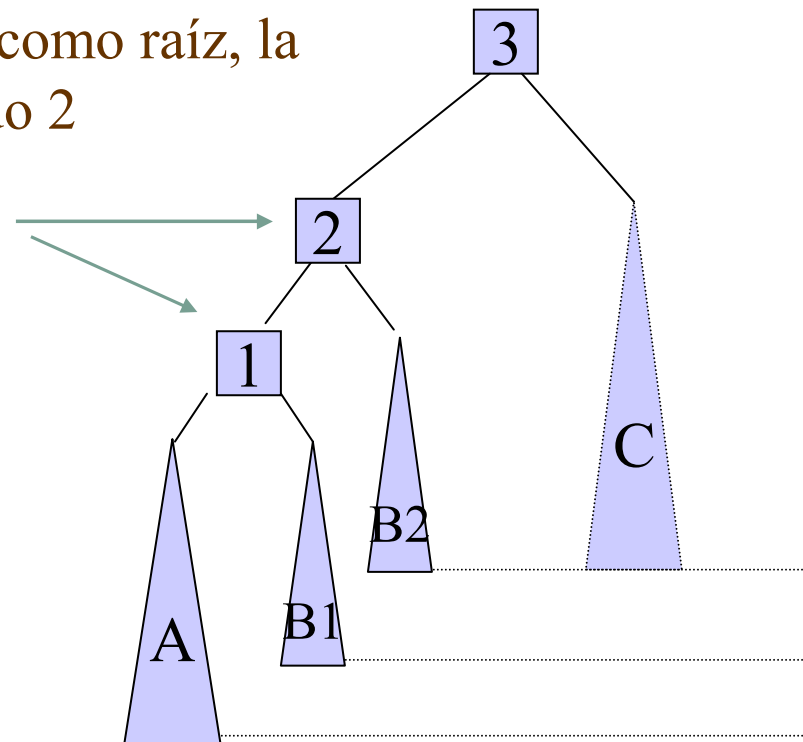
- Dado que el subárbol B tiene por lo menos un ítem, podemos considerar que está formado por una raíz y dos subárboles



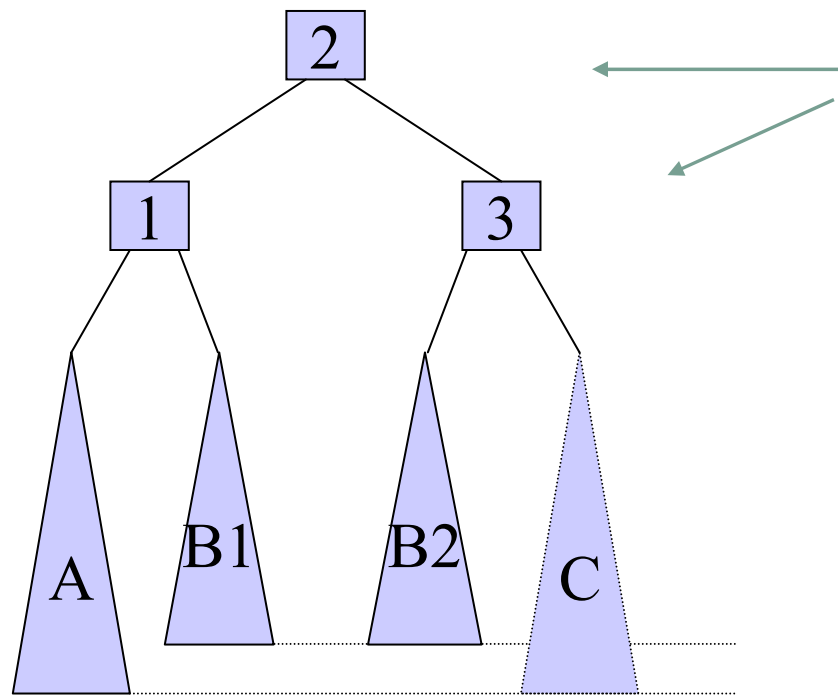
Rotación Doble (cont.)

- La rotación doble es similar a la simple, sólo que involucra cuatro subárboles en lugar de tres
- Ni los nodos 1 y 3 pueden quedar como raíz, la única alternativa es que quede el nodo 2

Primero se hace una rotación simple entre 1 y 2



Rotación Doble (cont.)



Luego se hace una rotación simple entre 2 y 3



Rotación Doble (cont.)

Caso 3: Rotación Doble Der-Izq

Es simétrica al caso 2, la inserción se produce en el subárbol izquierdo del hijo derecho.



Rotación Doble (cont.)

■ *Ejemplo:*

Continuar con el ejemplo anterior, insertando las claves del 8 al 15 en orden inverso.

!!! Tarea para el hogar !!!



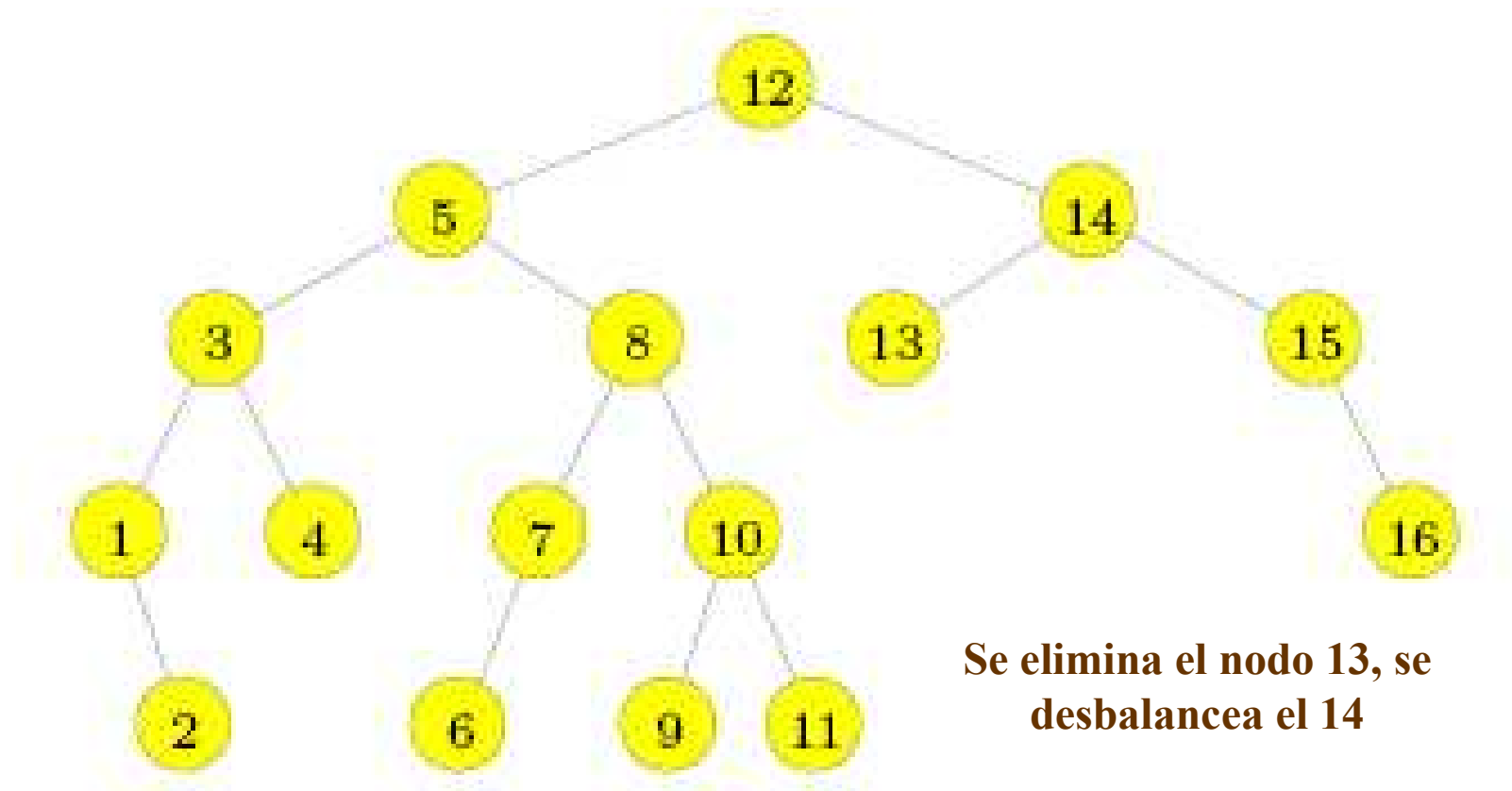
Eliminación de un nodo

- *La eliminación de un nodo es similar al borrado en un árbol binario de búsqueda.*
- *Luego de realizar el borrado se debe actualizar la altura de todos los nodos, desde el nodo en cuestión hasta la raíz.*
- *Puede destruirse la propiedad de balanceo*

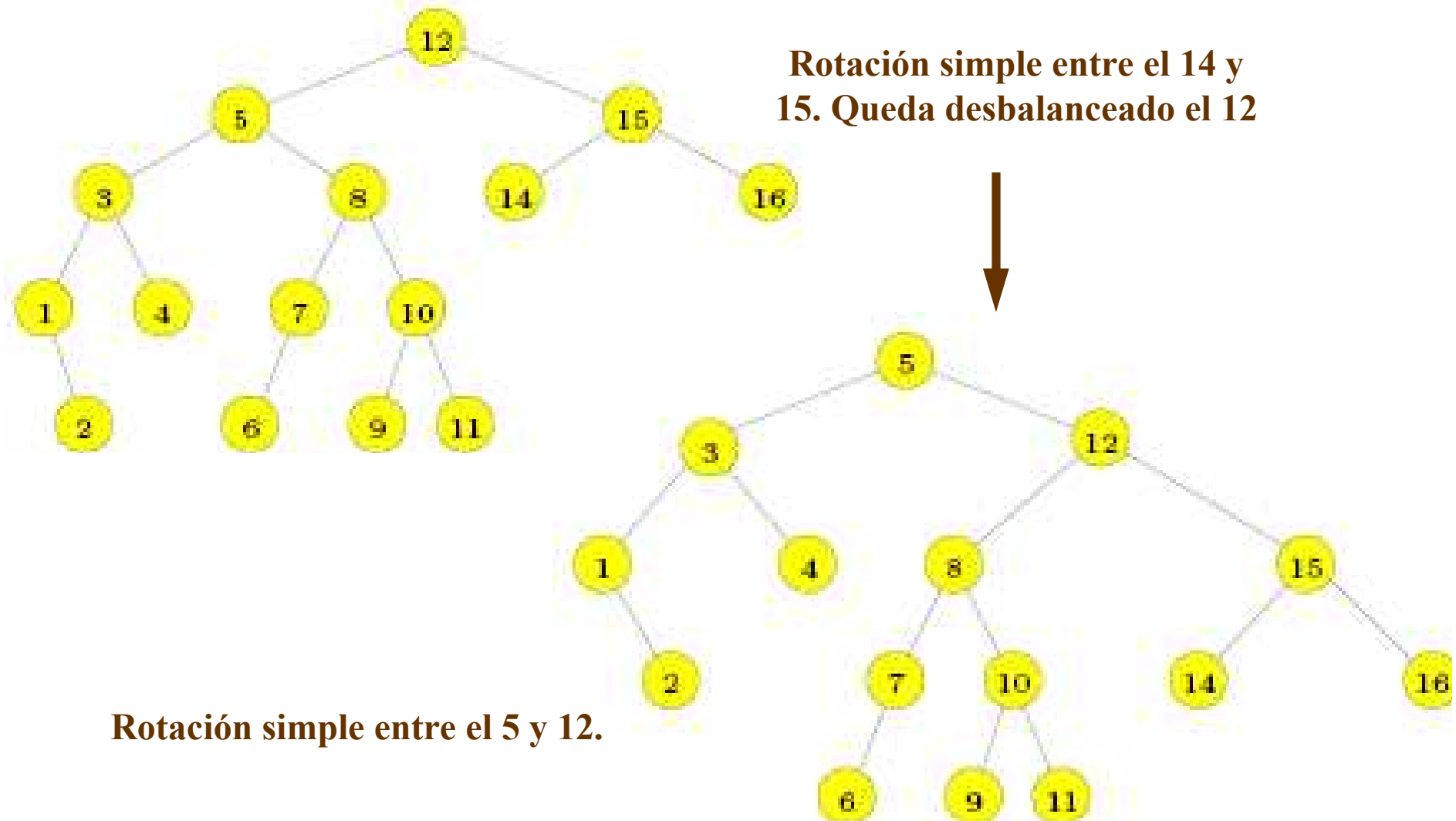



Rebalancear el árbol

Eliminación de un nodo



Eliminación de un nodo





Tiempo de ejecución de las operaciones en AVL

Las operaciones de:

Búsqueda

Inserción

Eliminación

Recorren la altura del árbol en el peor caso



Tiempo de ejecución de las operaciones en AVL (cont.)

Dado que un árbol AVL cumple con la condición de balanceo



Tomamos N_h como el número de nodos en un árbol AVL de altura h

$$\begin{aligned} N_h &\geq N_{h-1} + N_{h-2} + 1 \\ &\geq 2 N_{h-2} + 1 \\ &\geq 1 + 2(1 + 2 N_{h-4}) = 1 + 2 + 2^2 N_{h-4} \\ &\geq 1 + 2 + 2^2 + 2^3 N_{h-6} \\ &\dots \\ &\geq 1 + 2 + 2^2 + 2^3 + \dots + 2^{h/2} = 2^{(h/2+1)} - 1 \end{aligned}$$

Entonces:



Tiempo de ejecución de las operaciones en AVL (cont.)

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

$$h/2 + 1 \leq \log_2(n + 1)$$

$$h \leq 2(\log_2(n + 1) - 1)$$

Un análisis cuidadoso basado en la teoría de los números de Fibonacci, nos da un valor más ajustado de $1.44 \log_2(n + 2)$.



Operaciones sobre los árboles AVL: Conclusiones

- *Las operaciones de inserción y eliminación de un nodo son similares a las de un árbol binario de búsqueda.*
- *En ambas operaciones se debe actualizar la información de la altura y realizar rotaciones si es necesario.*
- *La inserción provoca una única reestructuración.*
- *La eliminación puede provocar varias reestructuraciones.*
- *Las operaciones son de $O(\log n)$*