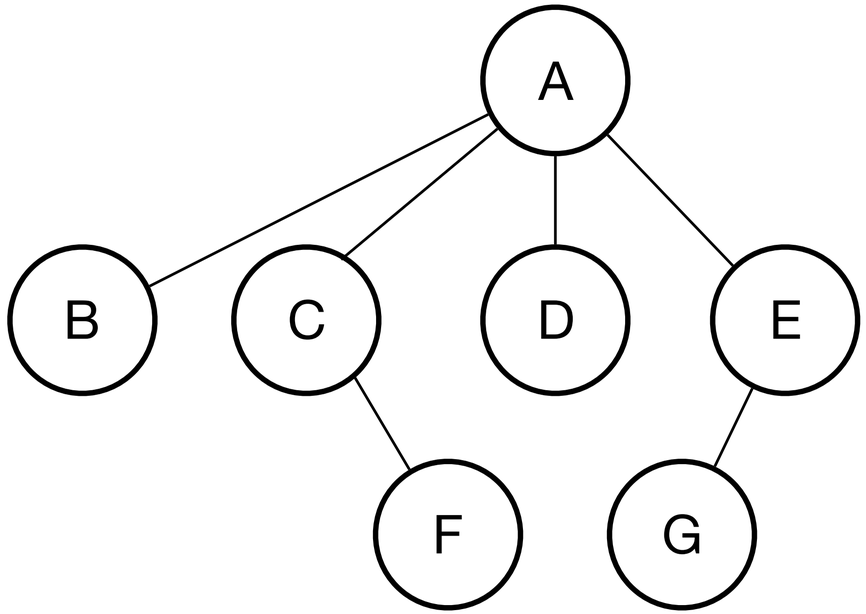
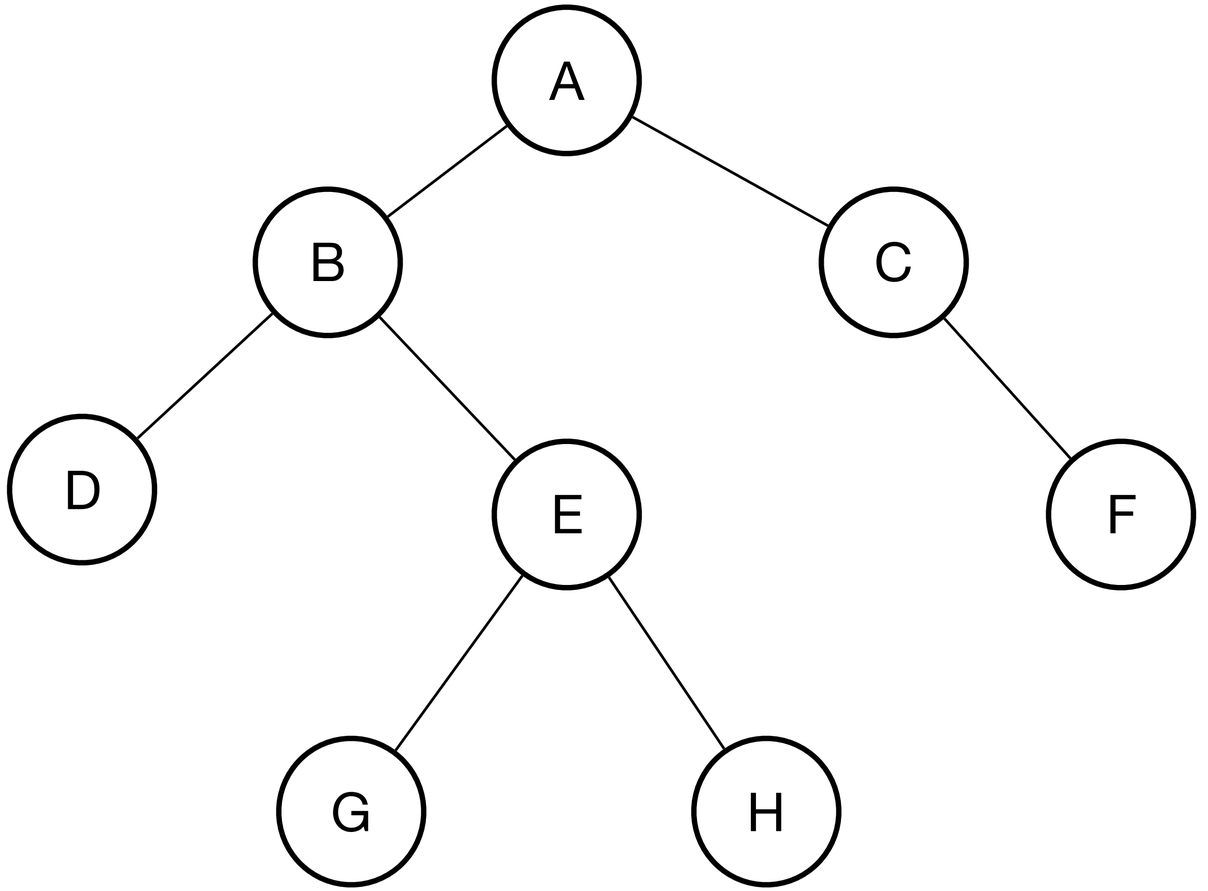
数据结构与算法实战
第四讲 树

4-1 二叉树及遍历

4.1.1 树与二叉树的定义

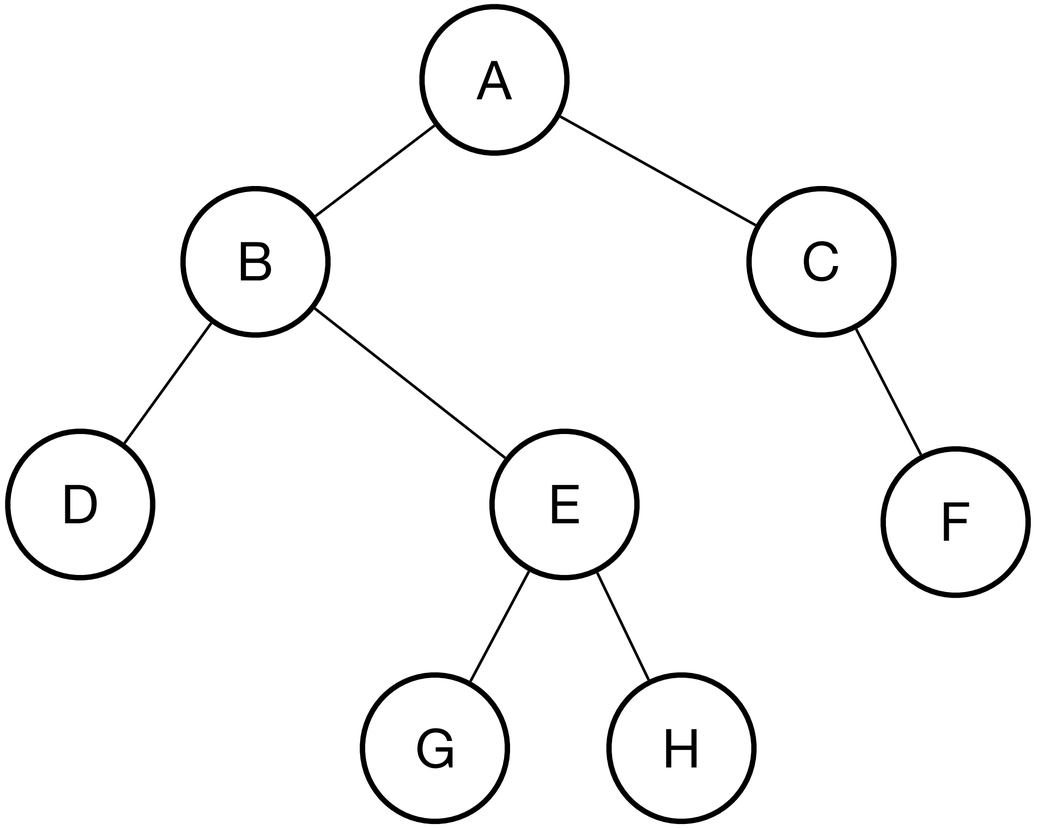


树



二叉树

4.1.1 树与二叉树的定义



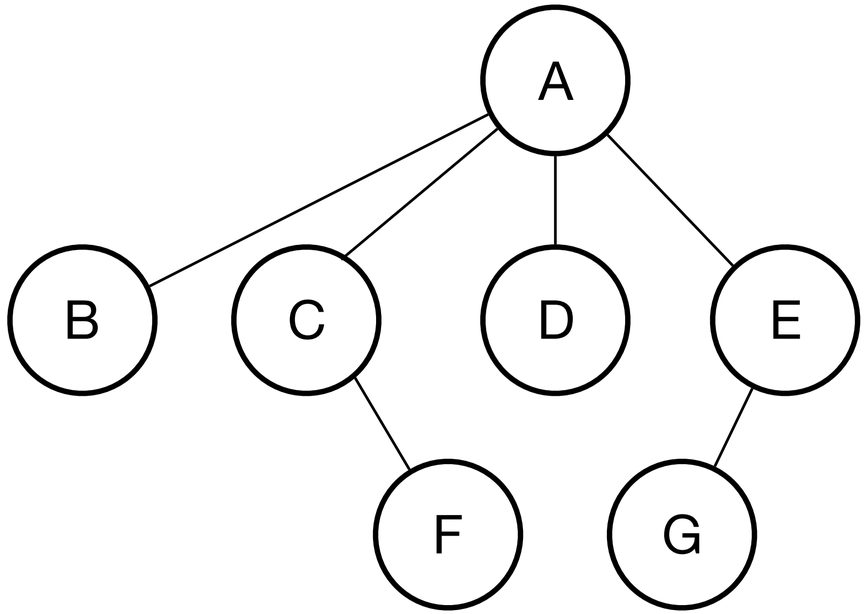
关于二叉树的几个概念：

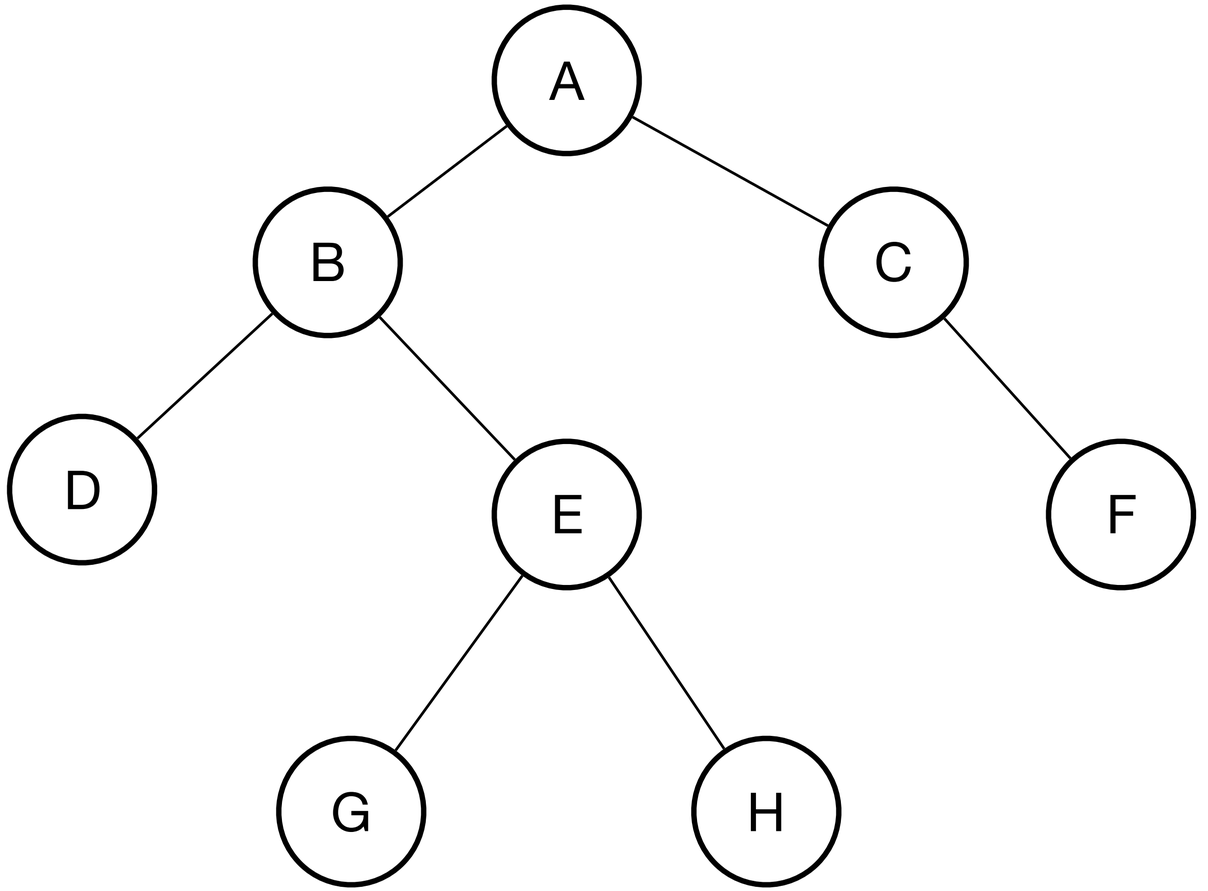
满二叉树（Full Binary Tree）

完全二叉树（Complete BT）

完美二叉树（Perfect BT）

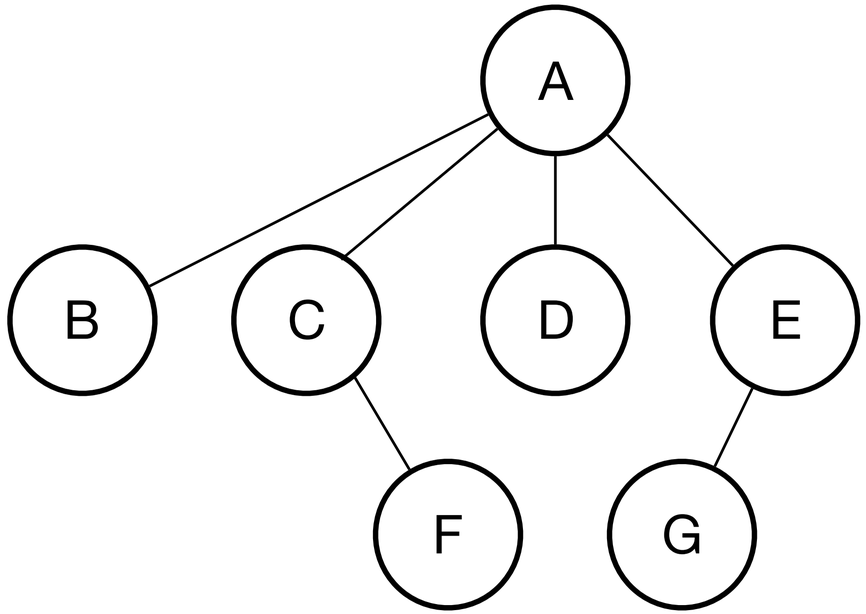
4.1.2 树与二叉树的顺序实现





以上两棵树如何用顺序实现（数组存储）？

4.1.3 树与二叉树的链式实现

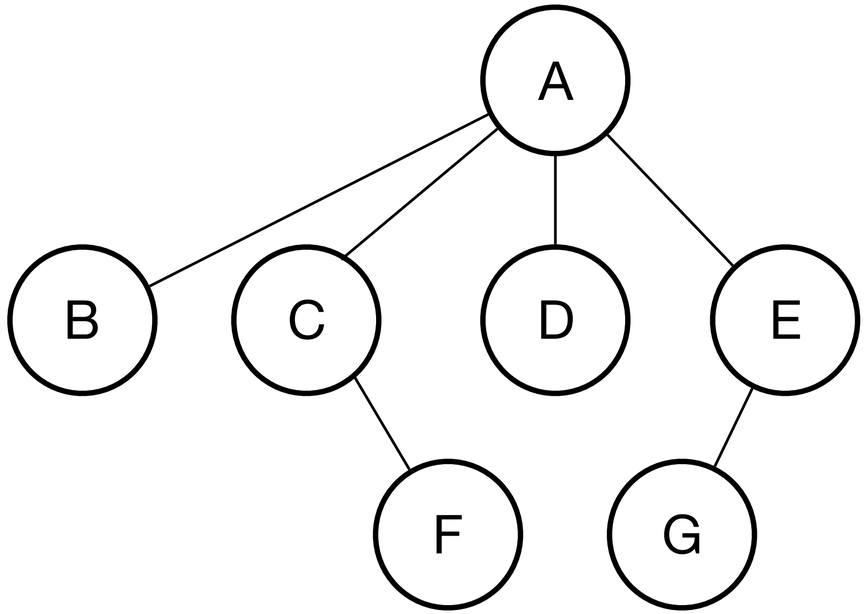


树

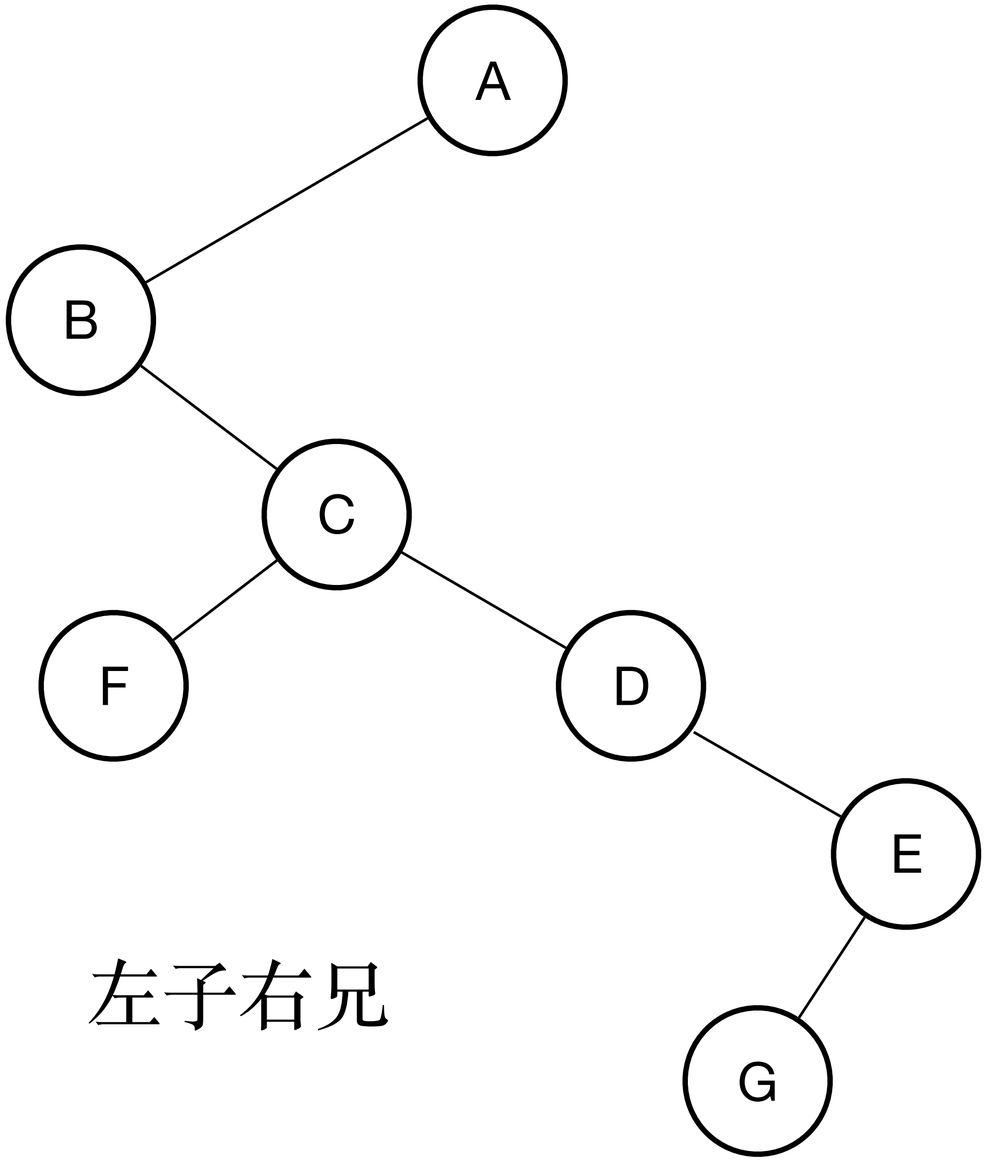
是否适合用链式存储?

节点的指针域指向子节点?

4.1.3 树与二叉树的链式实现

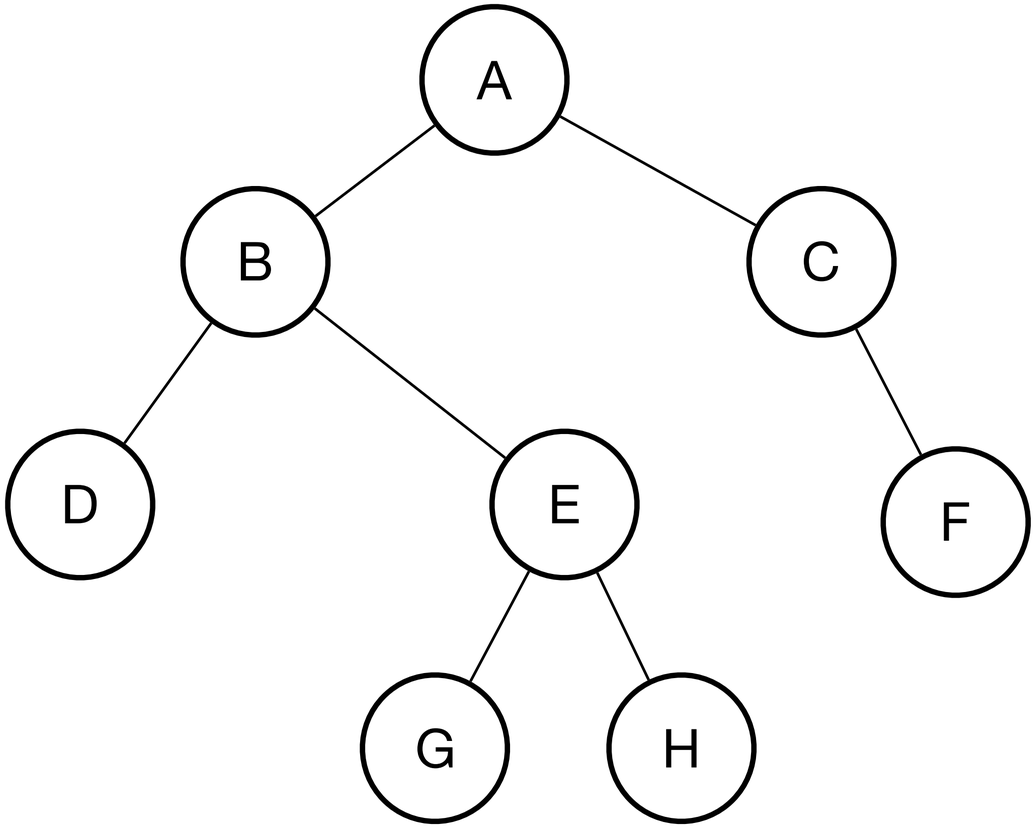


树



左子右兄

4.1.3 树与二叉树的链式实现



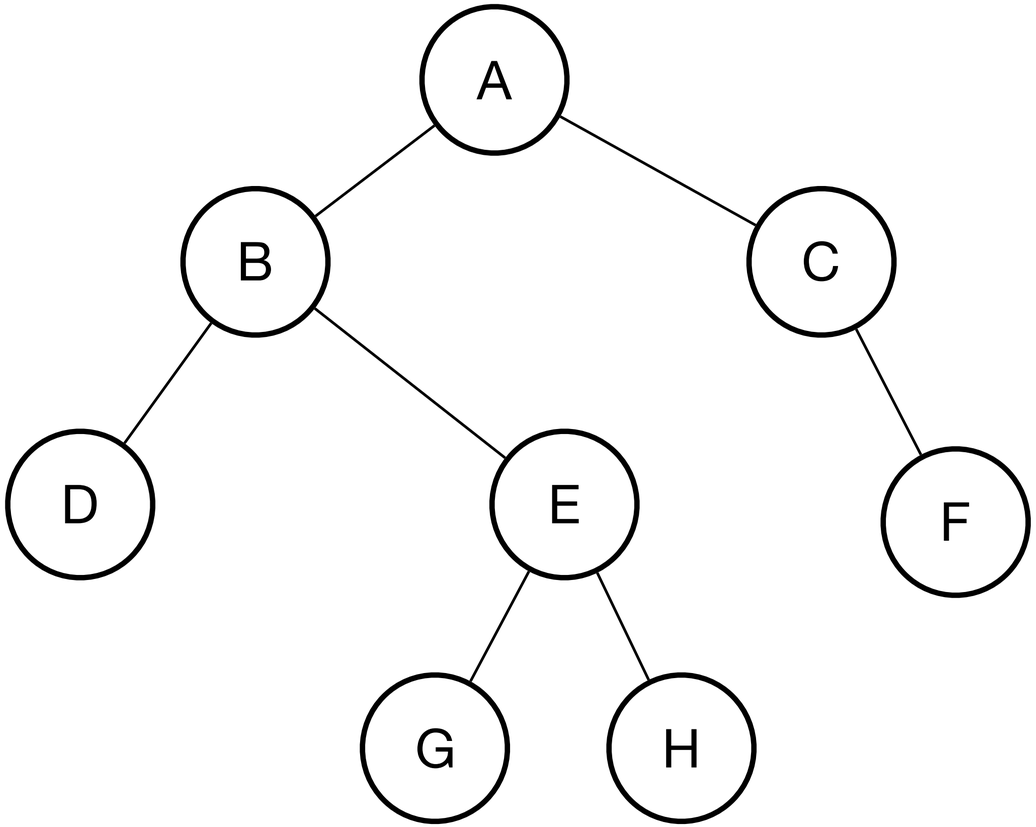
二叉树

每个节点有两个链?

left

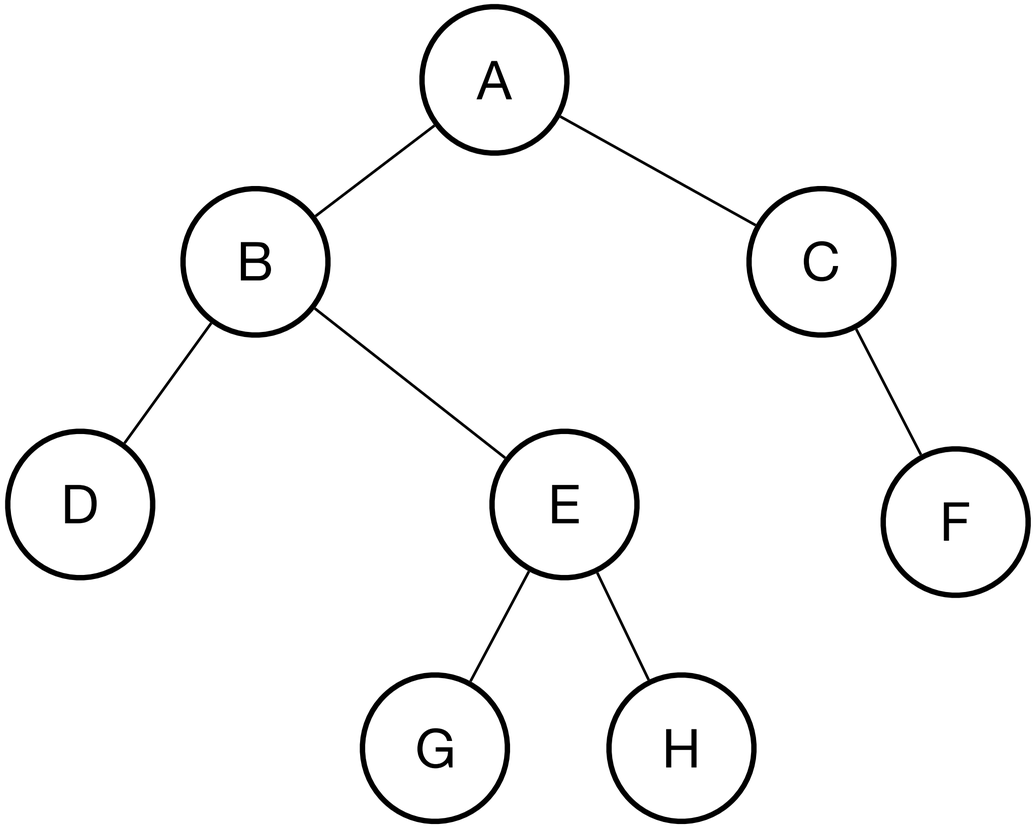
right

4.1.4 二叉树的遍历（递归实现）



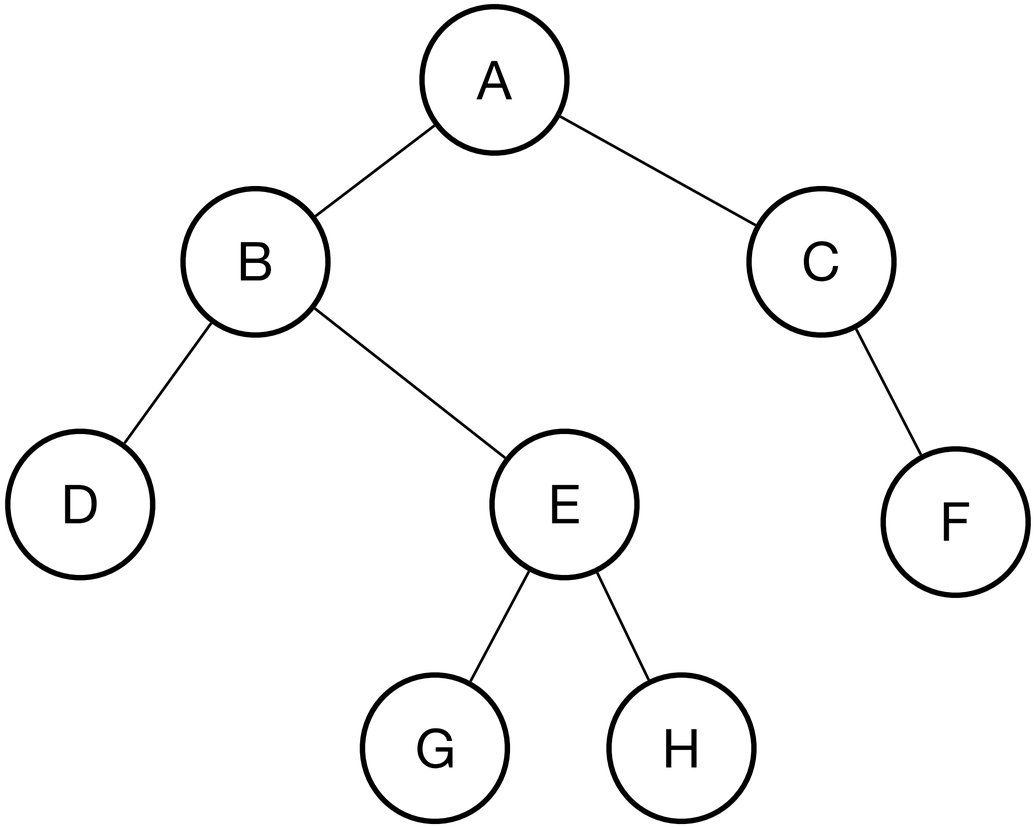
前序遍历：

4.1.4 二叉树的遍历（递归实现）



中序遍历：

4.1.4 二叉树的遍历（递归实现）

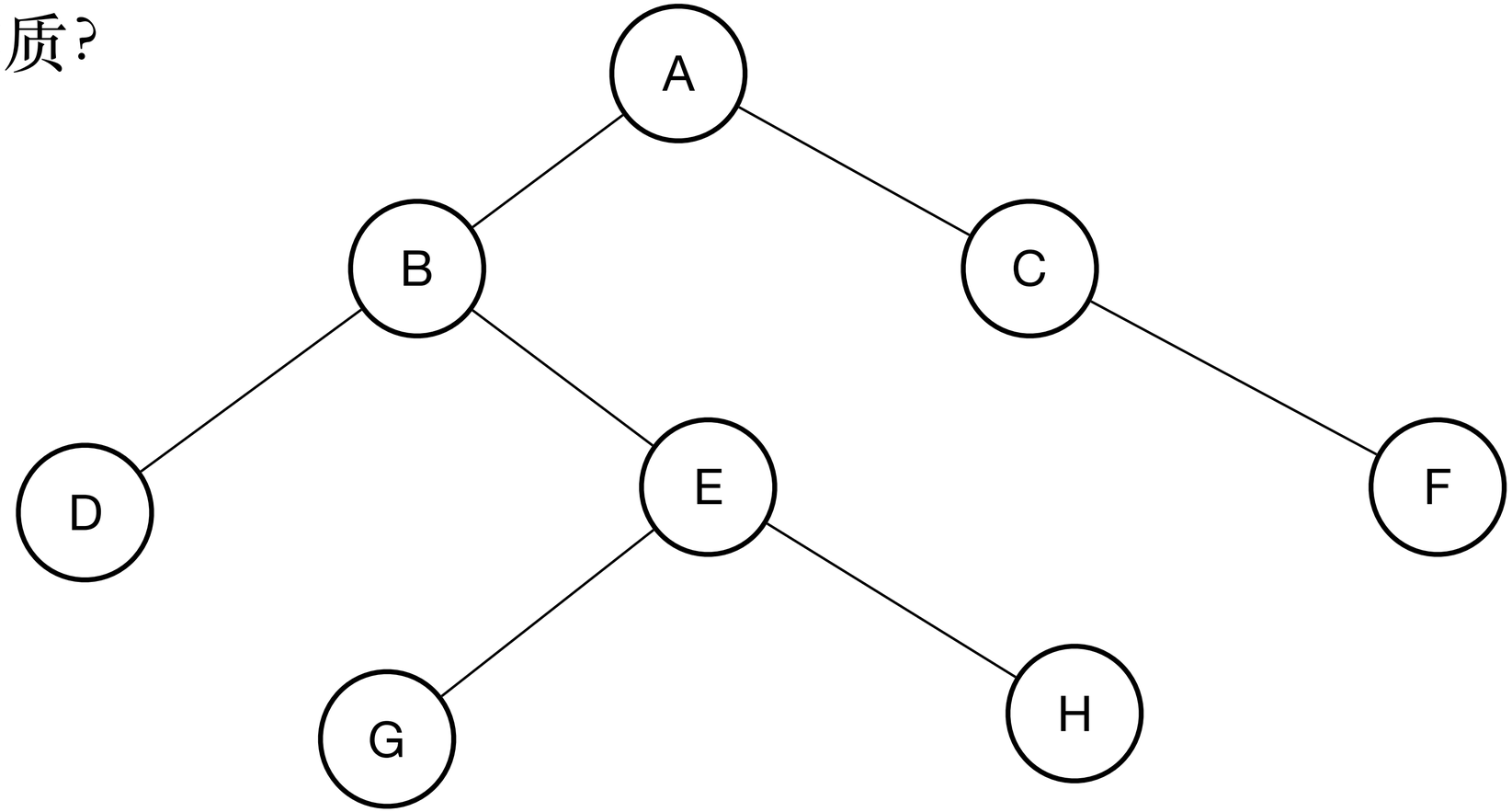


后序遍历：

4.1.5 二叉树的遍历（迭代实现）

三种遍历的实质?

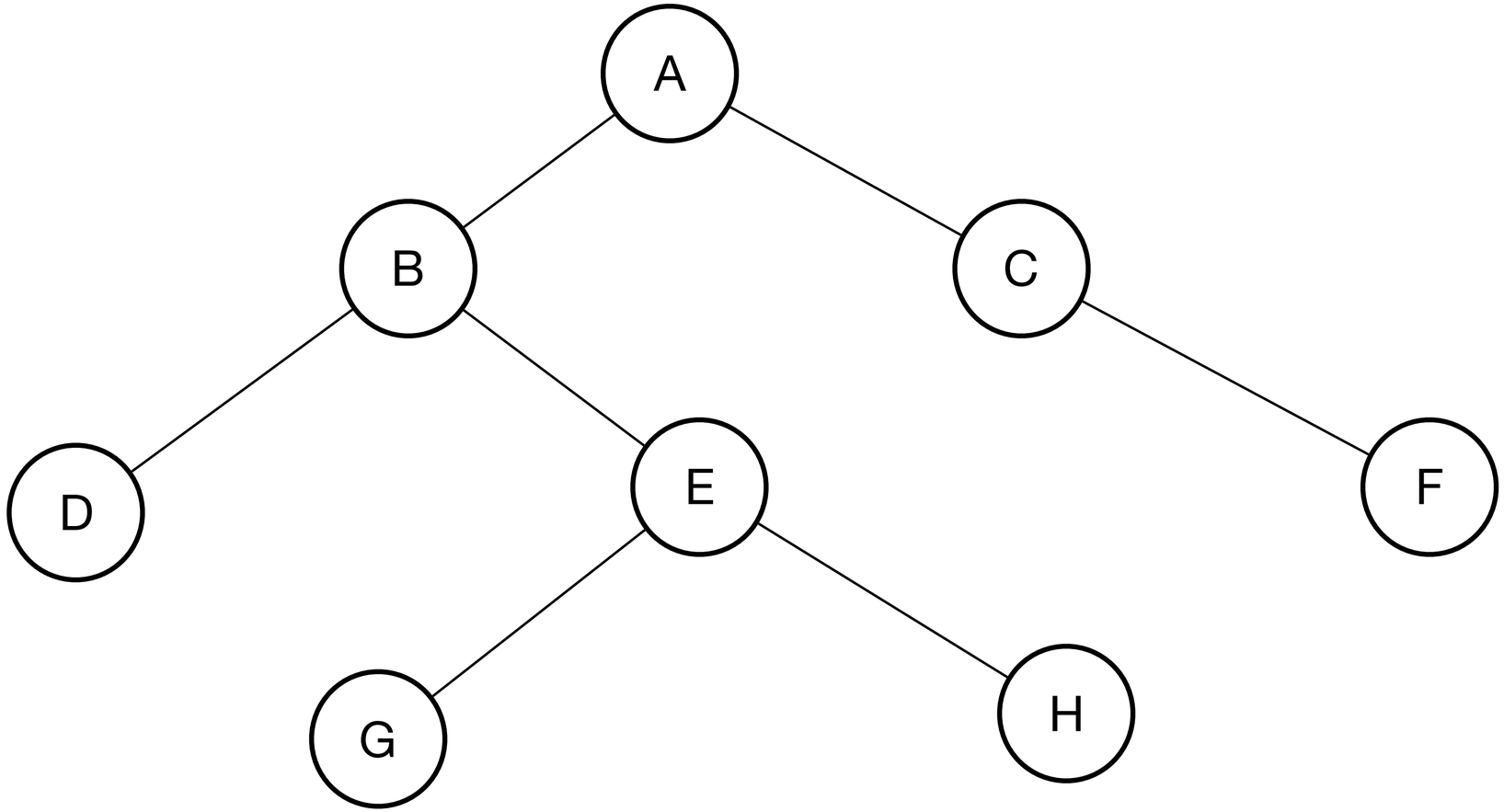
用什么实现?



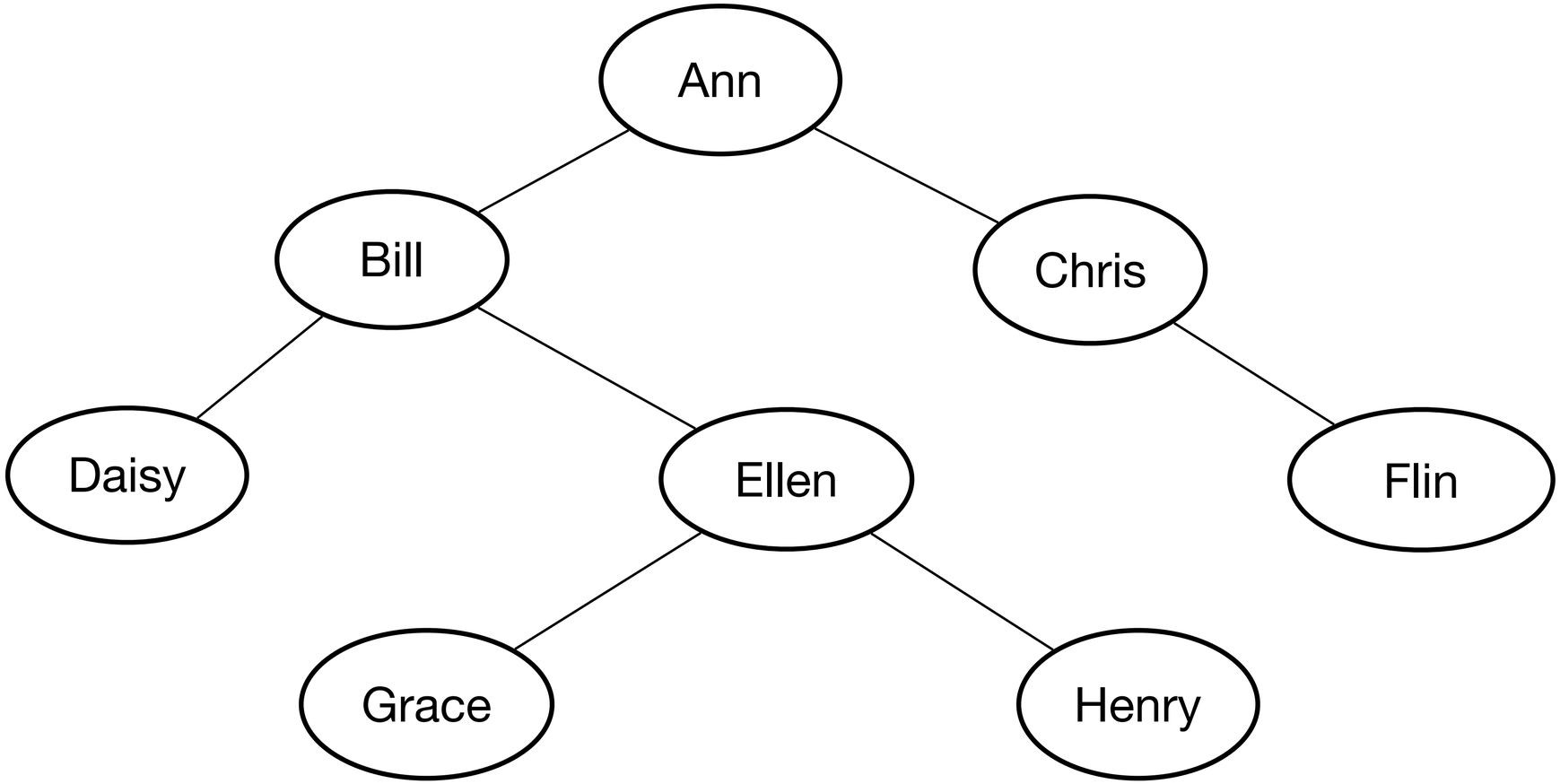
4.1.5 二叉树的遍历（迭代实现）

层序遍历

用什么实现?



4.1.6 实例：统计无后代的人数



4.1.6 实例：统计无后代的人数

输入格式:

Ann

Ann Bill Chris

Bill Daisy Ellen

Chris - Flin

Daisy - -

Ellen Grace Henry

Flin - -

Grace - -

Henry - -

-

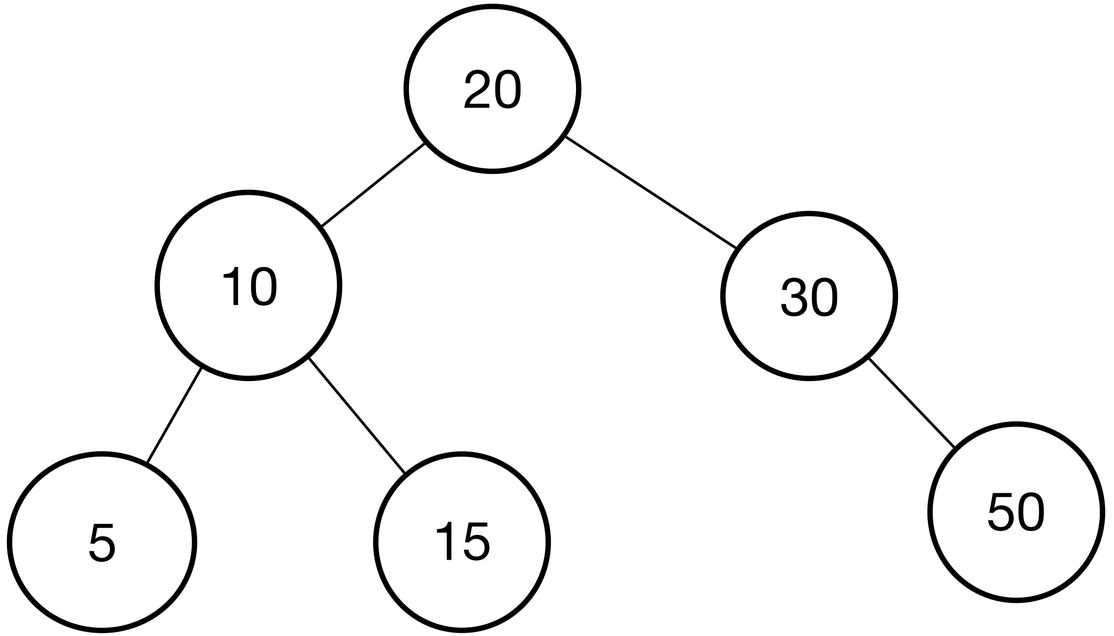
数据结构与算法实战
第四讲 树

4-2 二叉树搜索树（BST）

4.2.1 什么是二叉搜索树

二叉搜索树（Binary Search Tree）：

- 左子树元素比树根小
- 右子树元素比树根大
- 左右子树都是BST

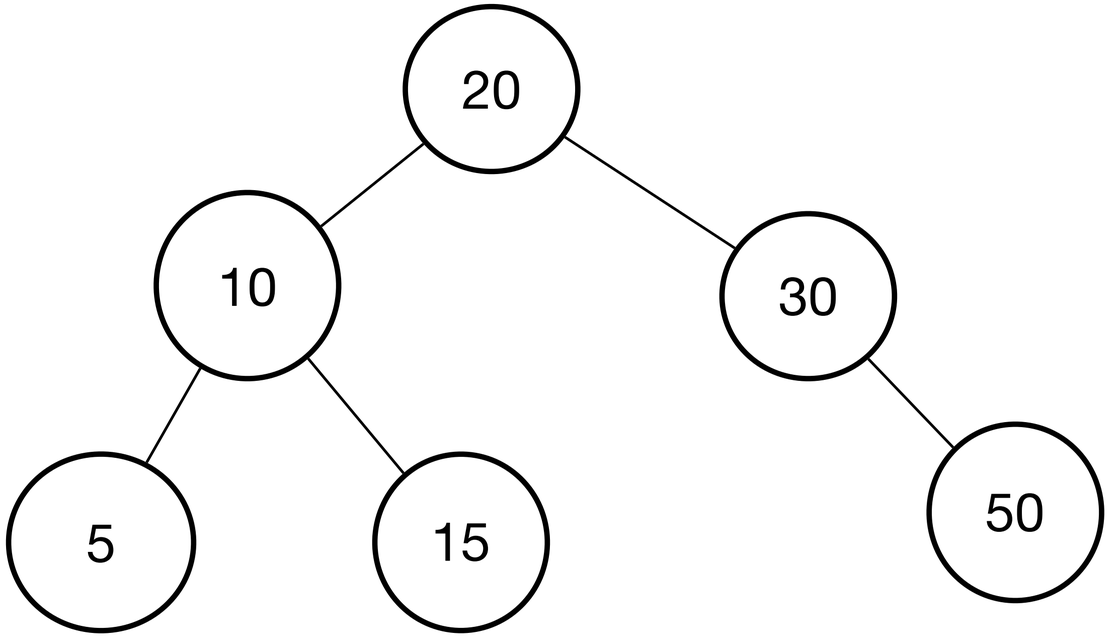


二叉树的查找效率？ BST呢？

4.2.2 BST的查找

查找操作:

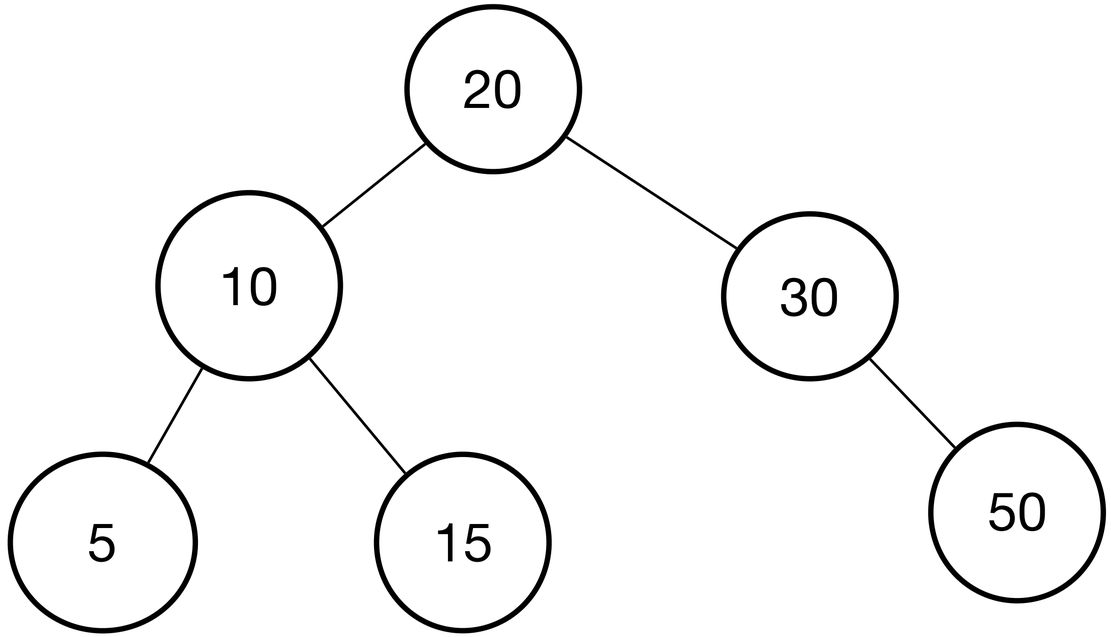
- FindMax
- FindMin
- FindX



4.2.3 BST的插入与删除

插入40:

插入10:

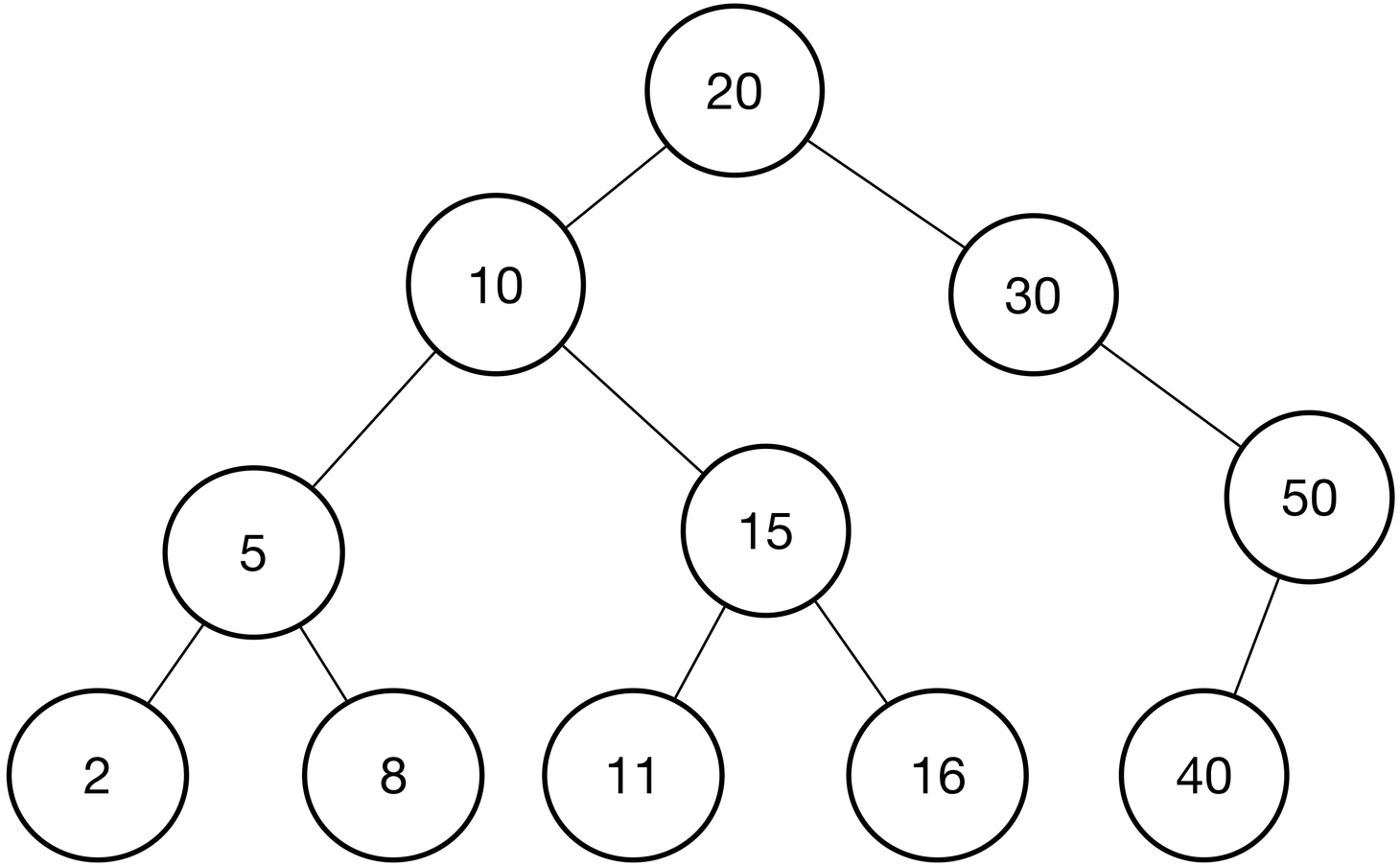


4.2.3 BST的插入与删除

删除40（叶子）：

删除30（单子节点）：

删除10/20（双子节点）：

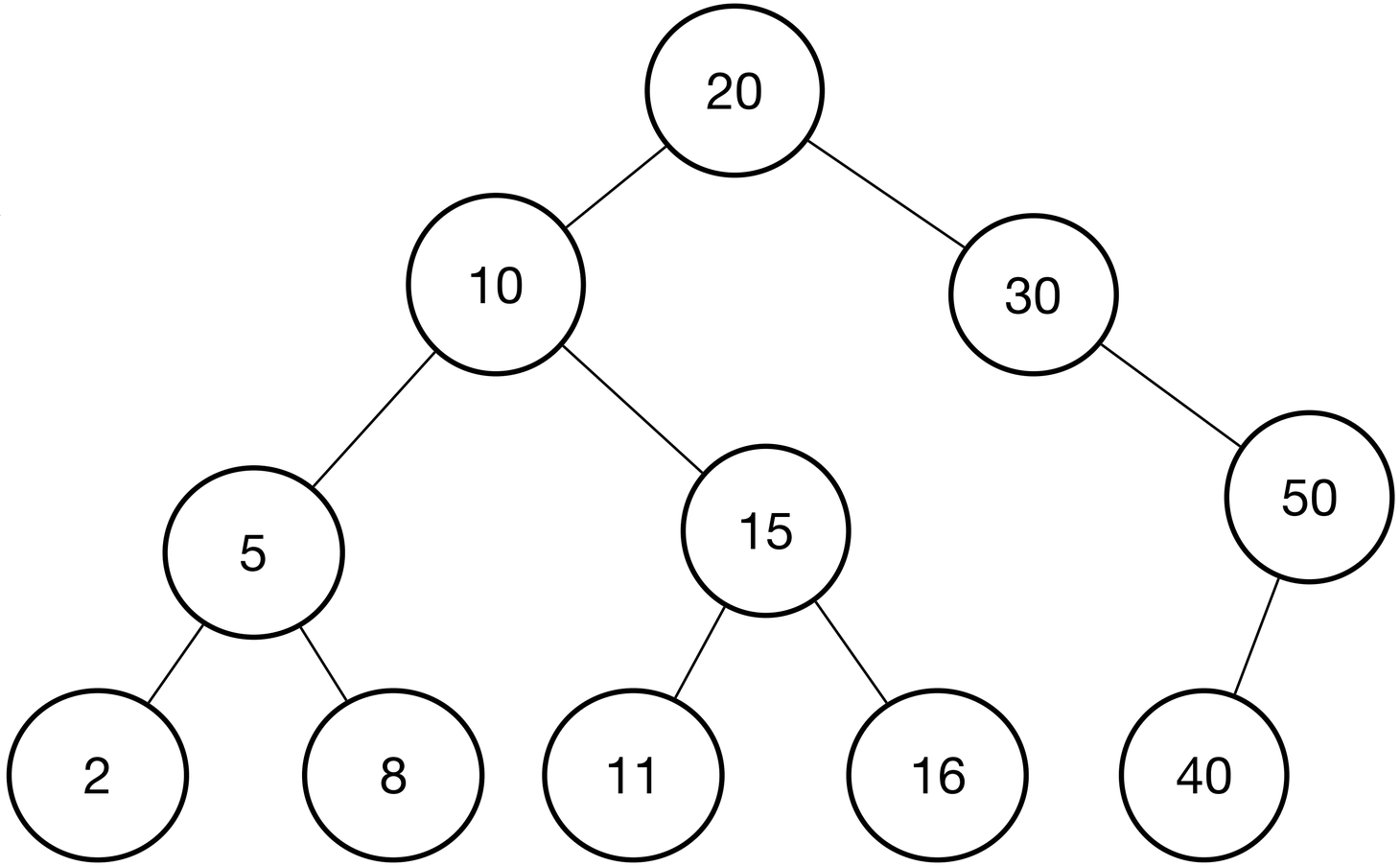


4.2.3 BST的插入与删除

删除10/20（双子节点）：

（1）用左子树最大或右子树最小元素代替待删节点

（2）删左子树最大（或右子树最小）



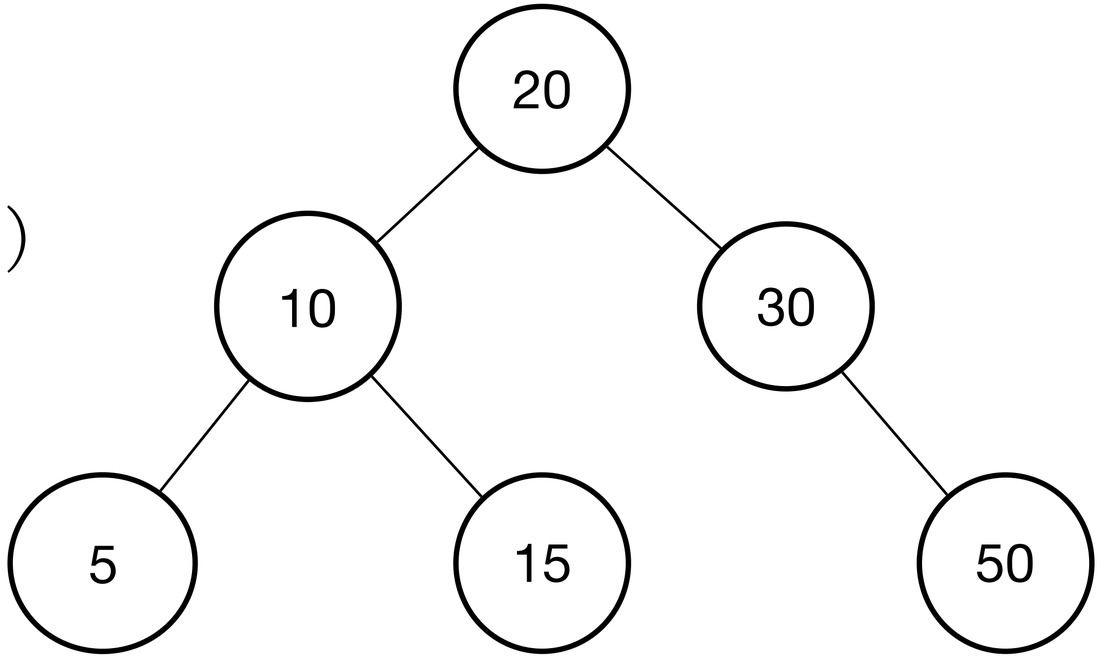
数据结构与算法实战
第四讲 树

4-3 平衡二叉搜索树（AVL树）

4.3.1 什么是AVL树

Adelson-Velsky-Landis Trees：

自平衡的二叉搜索树（self balanced BST）



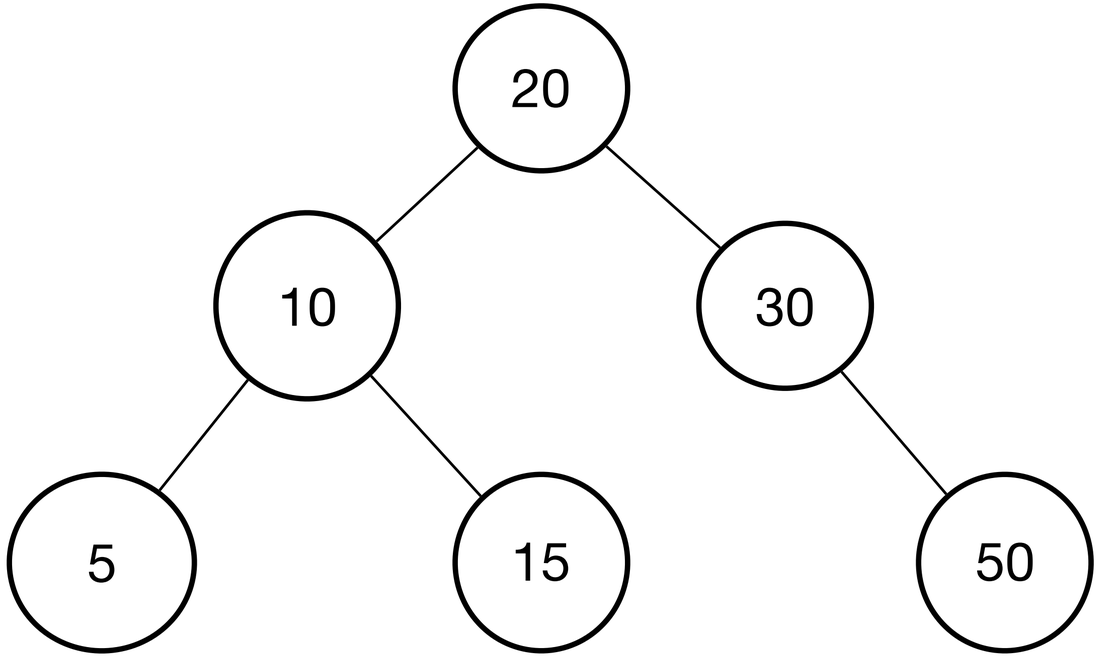
平衡：

- 空树是平衡的
- 非空树平衡 <=> 左右子树平衡 且 左右子树高度差绝对值<=1

=> 每个节点的平衡因子（balance factor）是 -1、 0 或 1

4.3.1 什么是AVL树

AVL树是BST



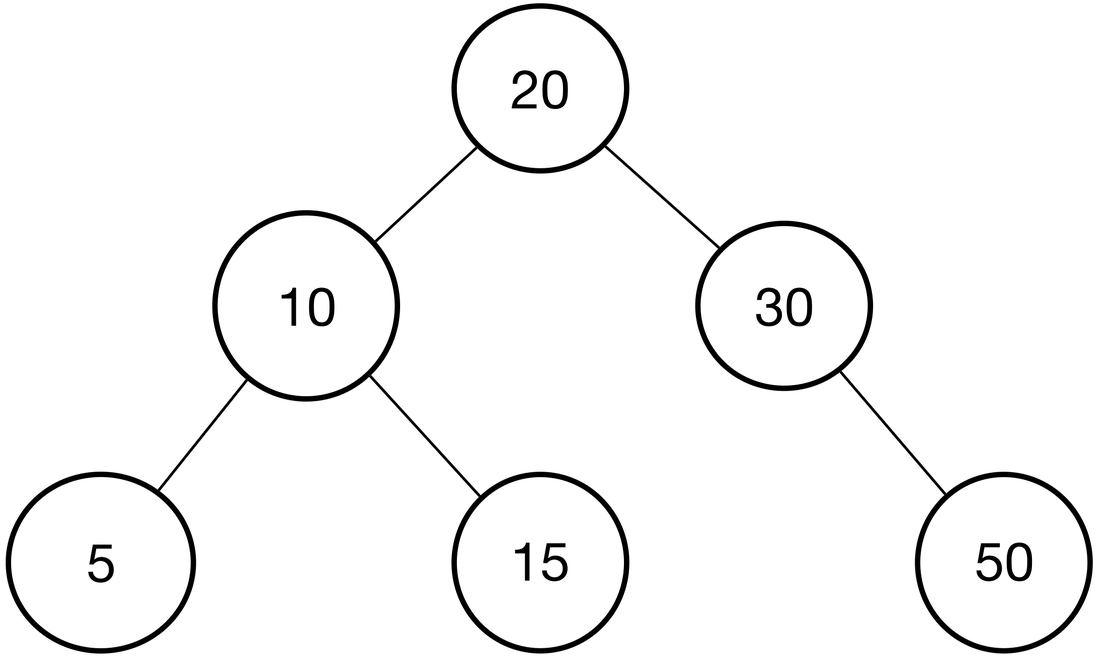
如何查找?

==> 与BST操作相同

查找效率?

4.3.1 什么是AVL树

AVL树是BST



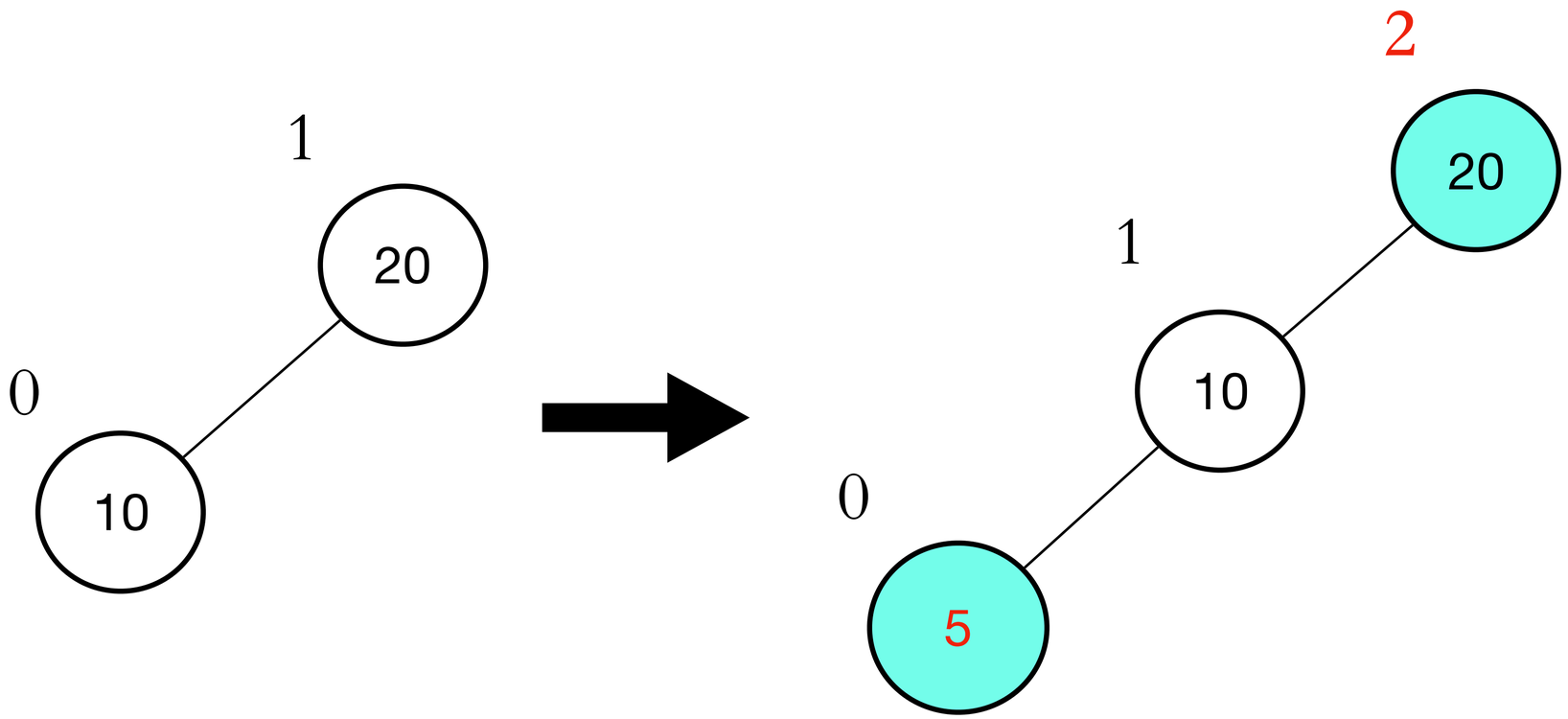
插入操作呢?

删除操作呢?

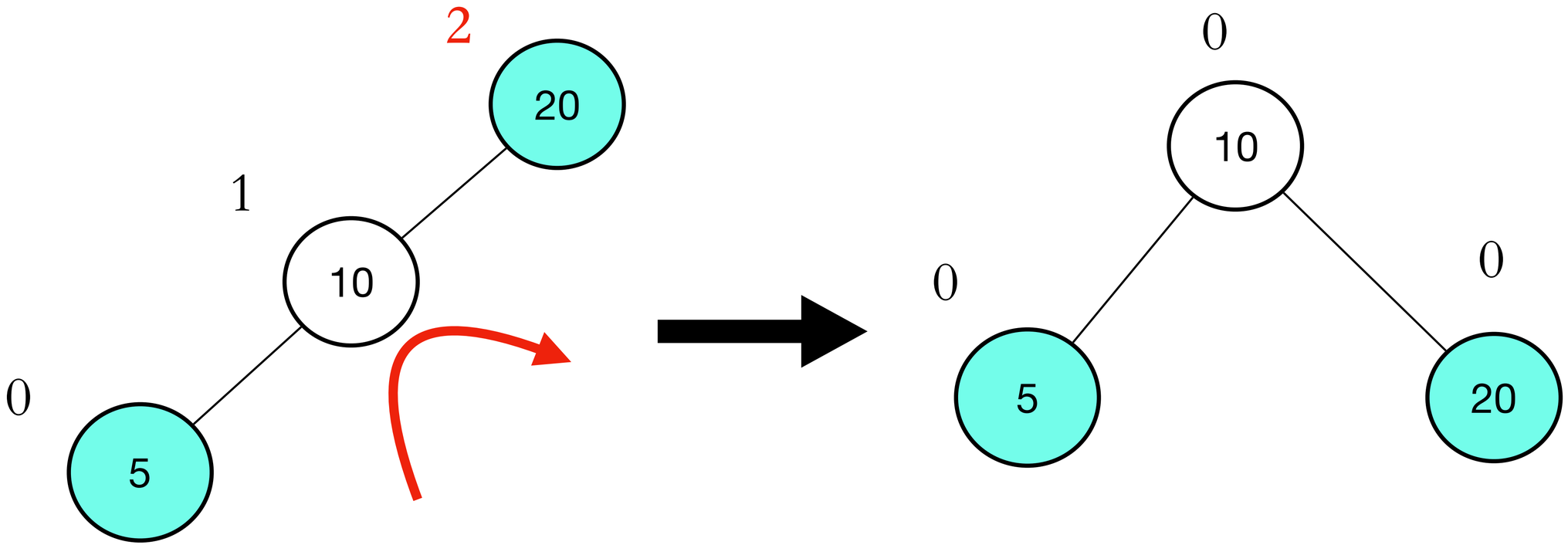
==> 与BST操作相同?

4.3.2 AVL树的插入

插入5:

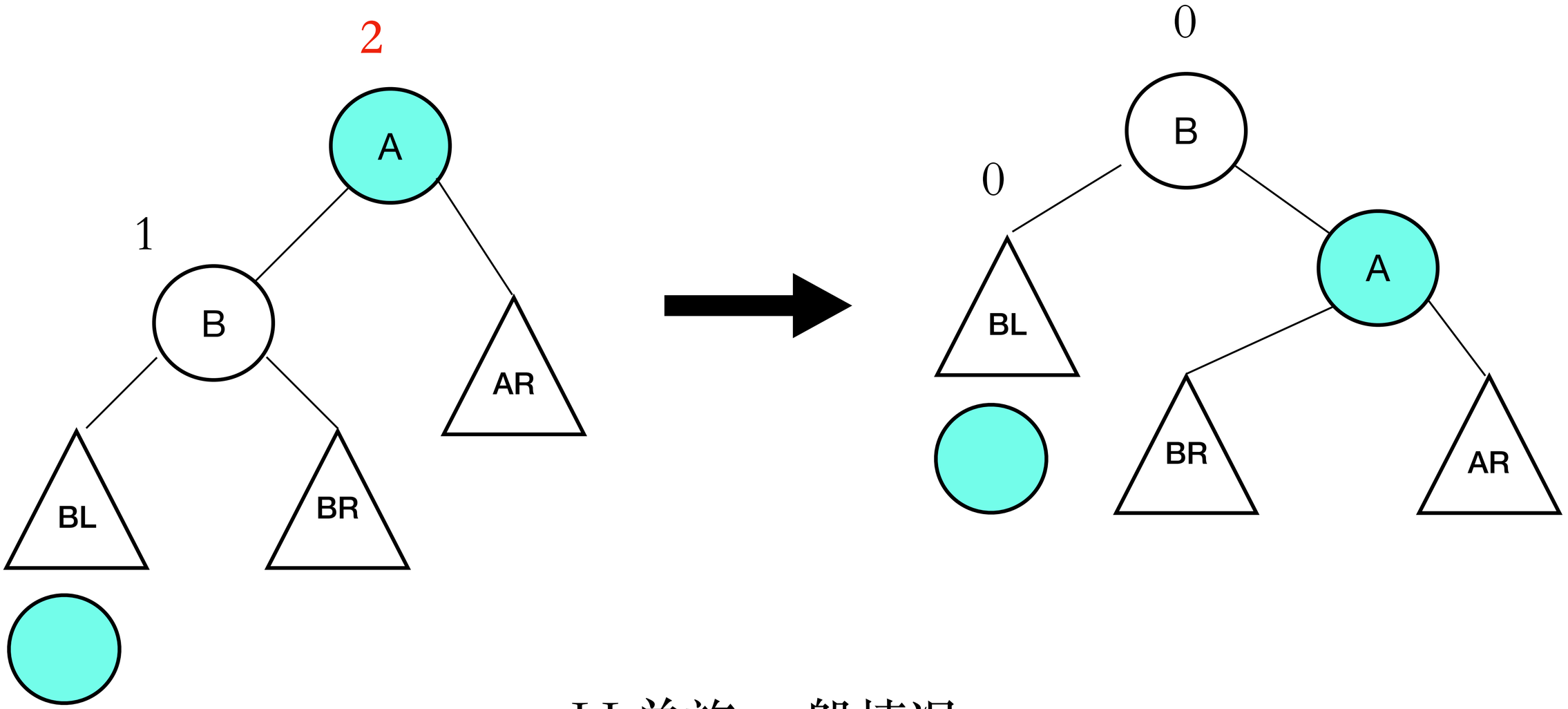


4.3.2 AVL树的插入



LL单旋 （左左单旋）

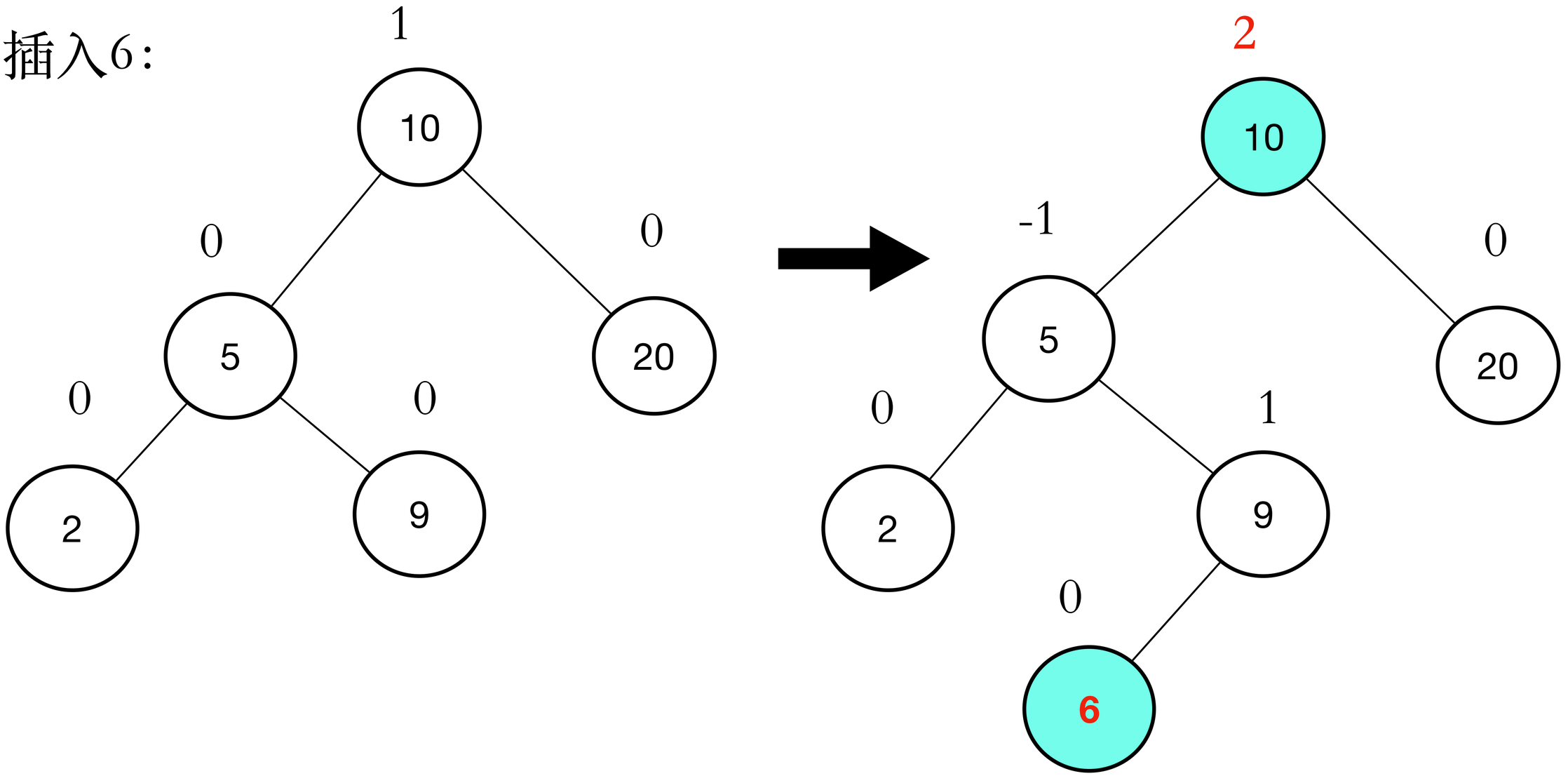
4.3.2 AVL树的插入



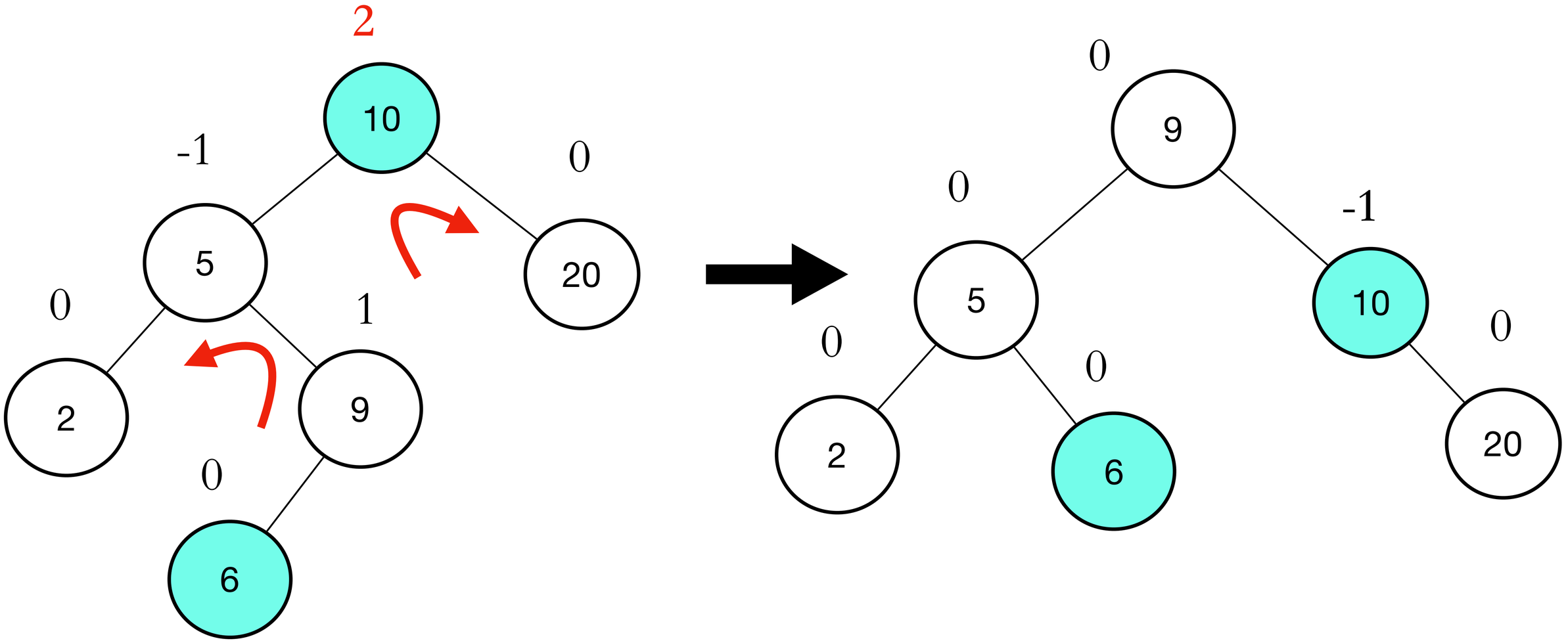
LL单旋 一般情况

4.3.2 AVL树的插入

插入6：

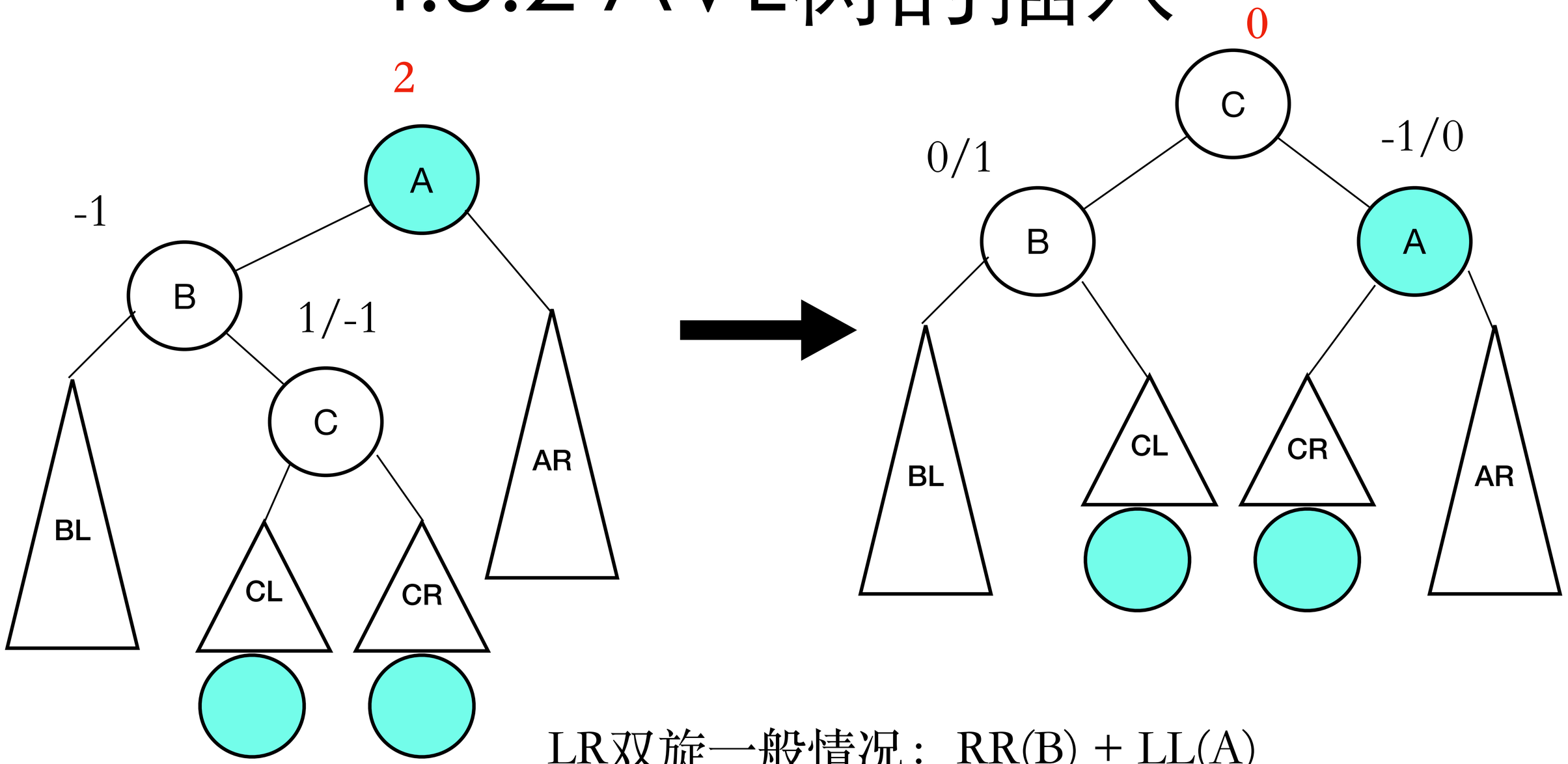


4.3.2 AVL树的插入



LR双旋（左右双旋）： RR(5) + LL(10)

4.3.2 AVL树的插入



LR双旋一般情况： RR(B) + LL(A)

4.3.2 AVL树的插入

RR单旋： 与LL单旋对称

RL双旋： 与LR双旋对称

4.3.3 AVL树的删除

lazy deletion（懒惰删除）