5.二叉树

中序遍历

观察

山中只见藤缠树

世上哪见树缠藤

青藤若是不缠树

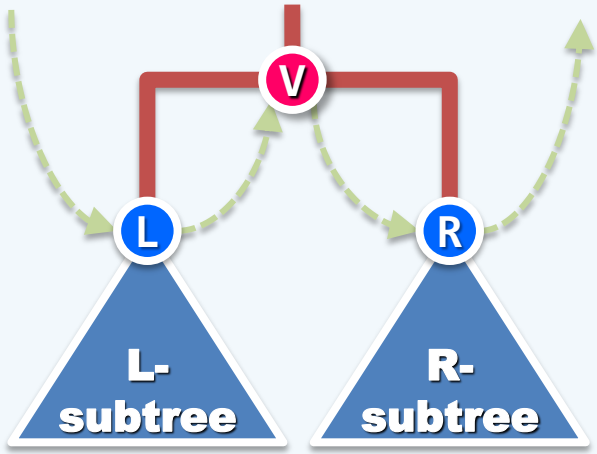
枉过一春又一春

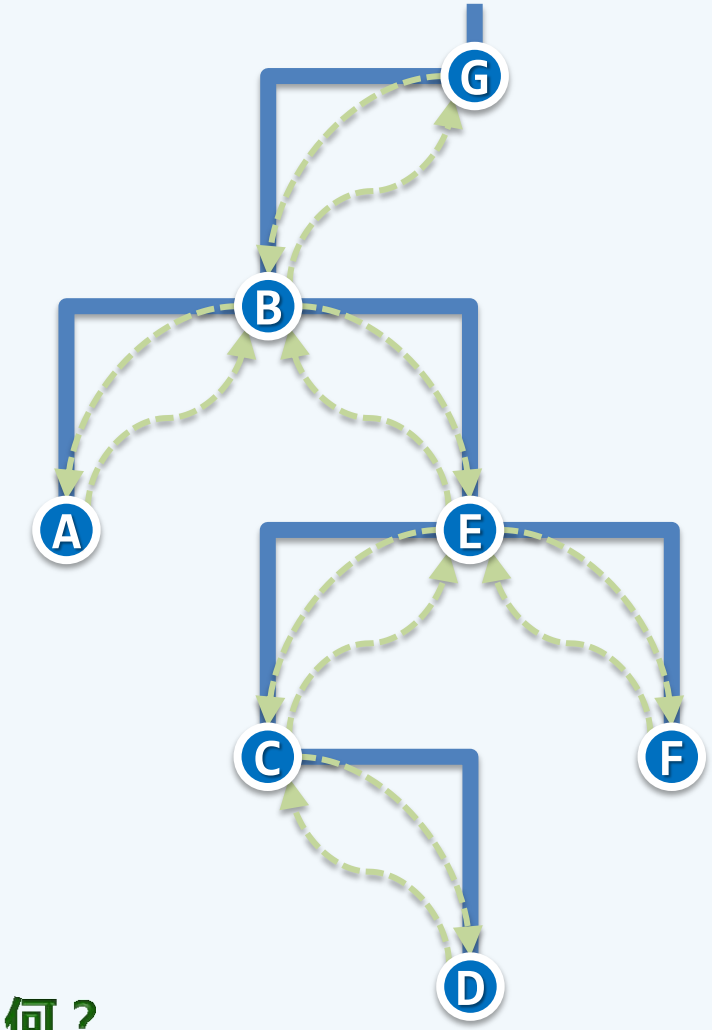
邓俊辉

deng@tsinghua.edu.cn

递归

```
template <typename T, typename VST>
void traverse( BinNodePosi(T) x, VST & visit ) {
    if ( !x ) return;
    traverse( x->lc, visit );
    visit( x->data );
    traverse( x->rc, visit );
} //T(n) = T(a) + O(1) + T(n-a-1) = O(n)
```





- 中序输出文件树结构：printBinTree()
- 挑战： 不依赖递归机制，能否实现中序遍历？如何实现？效率如何？

难点

❖ 难度在于

尽管右子树的递归遍历是尾递归，但左子树却严格地不是

❖ 解决方法

找到第一个被访问的节点　//仿照迭代的先序遍历算法

将其祖先用栈保存　//按照被访问过程的逆序

❖ 这样，原问题就被分解为

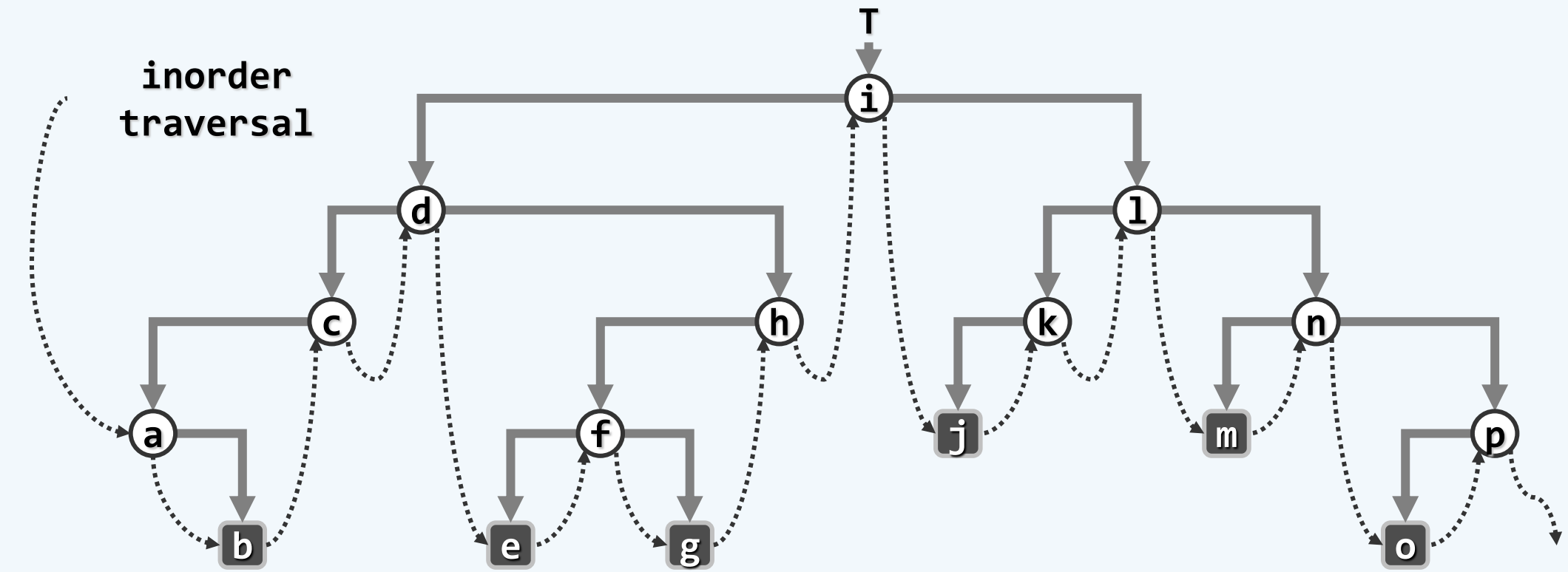
依次对若干棵右子树的遍历问题　//依什么“次”？

❖ 于是，首先要解决的问题就是：

中序遍历任一二叉树T时

首先被访问的是哪个节点？如何找到它？

观察



→ a → b → c → d → e → f → g → h → i → j → k → l → m → n → o → p →

藤缠树

❖ 从根出发沿左分支下行，直到最深的节点

——它就是全局首先被访问者

❖ 从宏观上，整个遍历过程可划分为d+1步迭代

访问L_k，再遍历T_k，k = d, ..., 2, 1, 0

❖ 不同右子树的遍历

相互独立

自成一个子任务

endpoint of the vine