05-G1

二叉树

后序遍历：观察

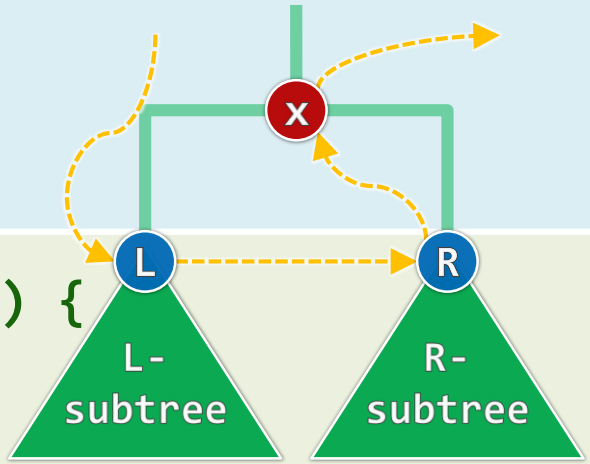
看了，因向仙姑道：“敢烦仙姑引我到那各司中游玩游玩，不知可使得？”仙姑道：“此各司中皆贮的是普天之下所有的女子过去未来的簿册，尔凡眼尘躯，未便先知的。”

邓俊辉
deng@tsinghua.edu.cn

后序实例：子树删除

```
template <typename T> Rank BinTree<T>::remove( BinNodePosi<T> x ) {
   FromParentTo( * x ) = NULL;
   updateHeightAbove( x->parent ); //更新祖先高度（其余节点亦不变）
   Rank n = removeAt(x); _size -= n; return n;
}
```

postorder




```
template <typename T> static Rank removeAt( BinNodePosi<T> x ) {
   if ( ! x ) return 0;
   Rank n = 1 + removeAt( x->lc ) + removeAt( x->rc );
   release(x->data); release(x); return n;
}
```

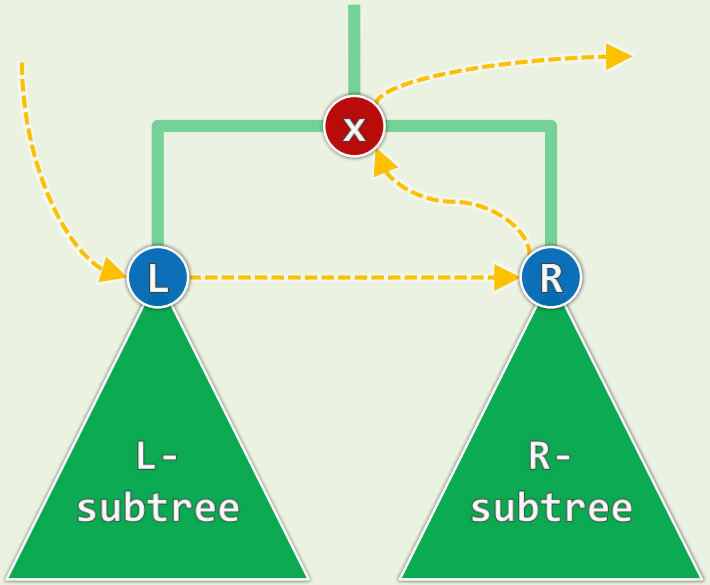
递归实现

❖ 应用：BinNode::size() + BinTree::updateHeight()

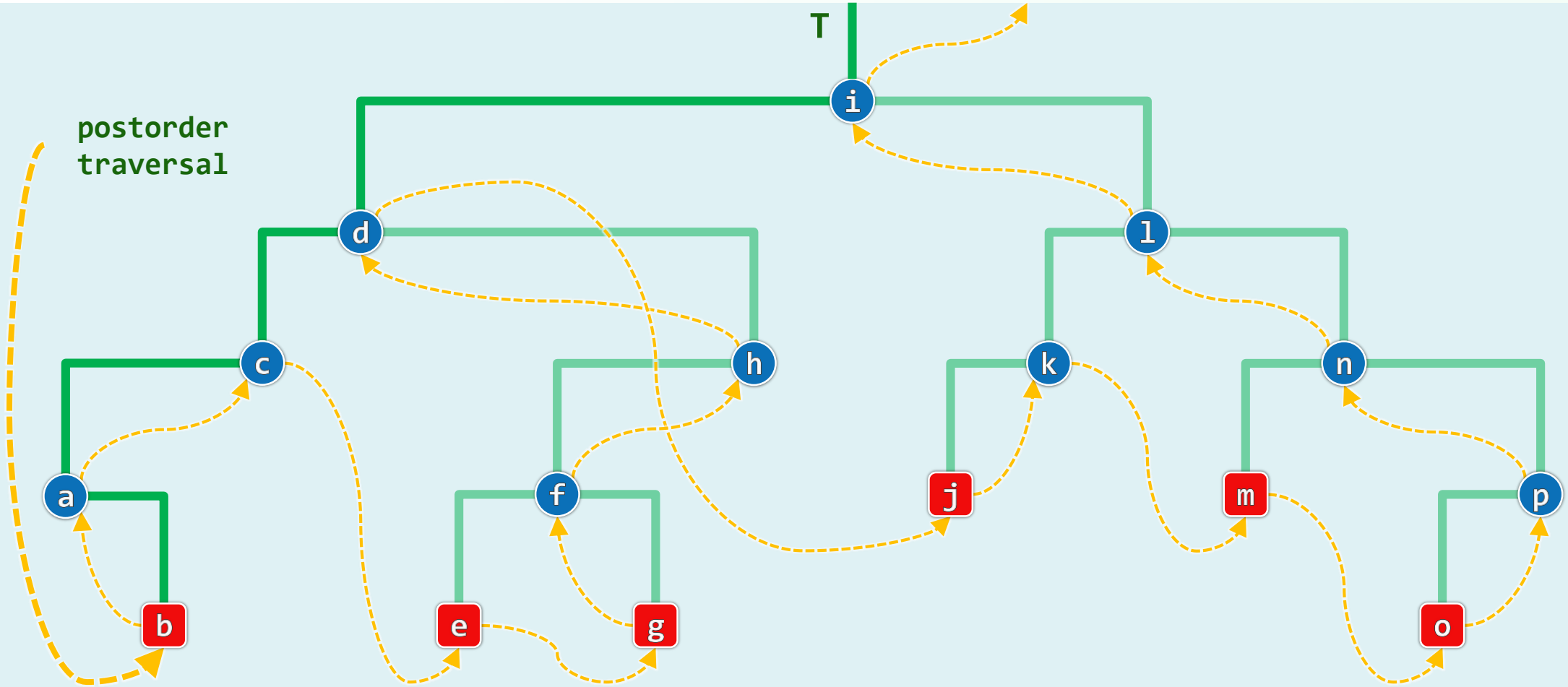
❖
```
template <typename T, typename VST>
void traverse( BinNodePosi<T> x, VST & visit ) {
    if ( ! x ) return;
    traverse( x->lc, visit );
    traverse( x->rc, visit );
    visit( x->data );
}
```

❖ $T(n) = T(a) + T(n - a - 1) + \mathcal{O}(1) = \mathcal{O}(n)$

❖ 挑战：不依赖递归机制，如何实现后序遍历？效率如何？



观察

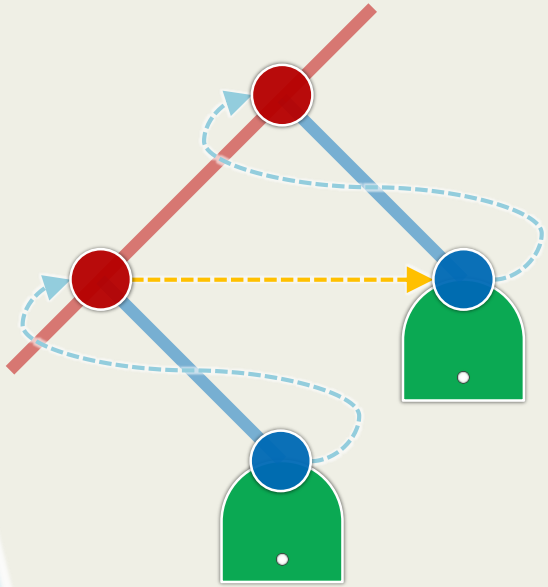

T
postorder
traversal
i
d
l
c
h
k
n
a
f
j
m
p
b
e
g
o

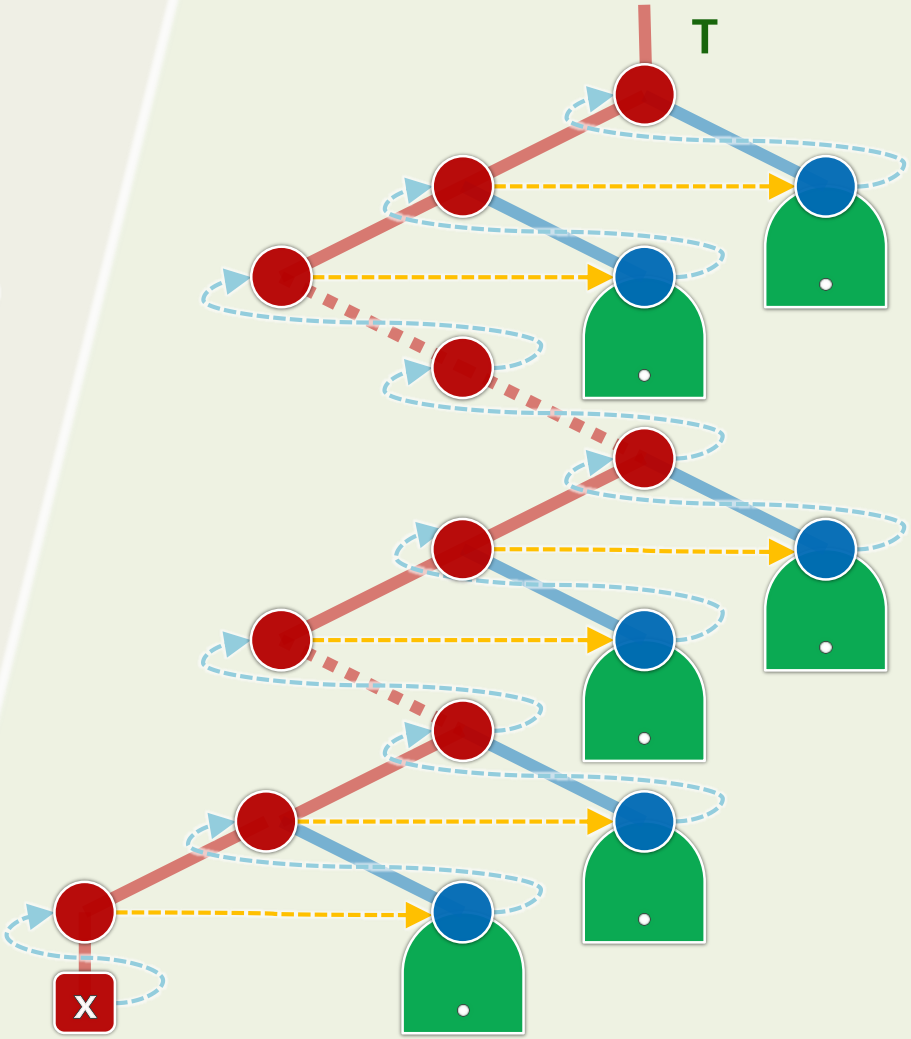

b
a
c
e
g
f
h
d
j
k
m
o
p
n
l
i

藤缠树

- 从根出发下行
  - 尽可能沿左分支
  - 实不得已，才沿右分支
- 最后一个节点
  - 必是叶子，而且
  - 是按中序遍历次序最靠左者

  也是递归版中visit()首次执行处
- 这片叶子，将首先接受访问...





leftmost leaf