二叉搜索树

AVL树：插入

06-03

名不正则言不顺，言不顺则事不成

没有什么是一次旋转解决不了的

如果有，那就两次

邓俊辉

deng@tsinghua.edu.cn

单旋：黄色方块恰好存在其一

❖ 同时可有多个失衡节点

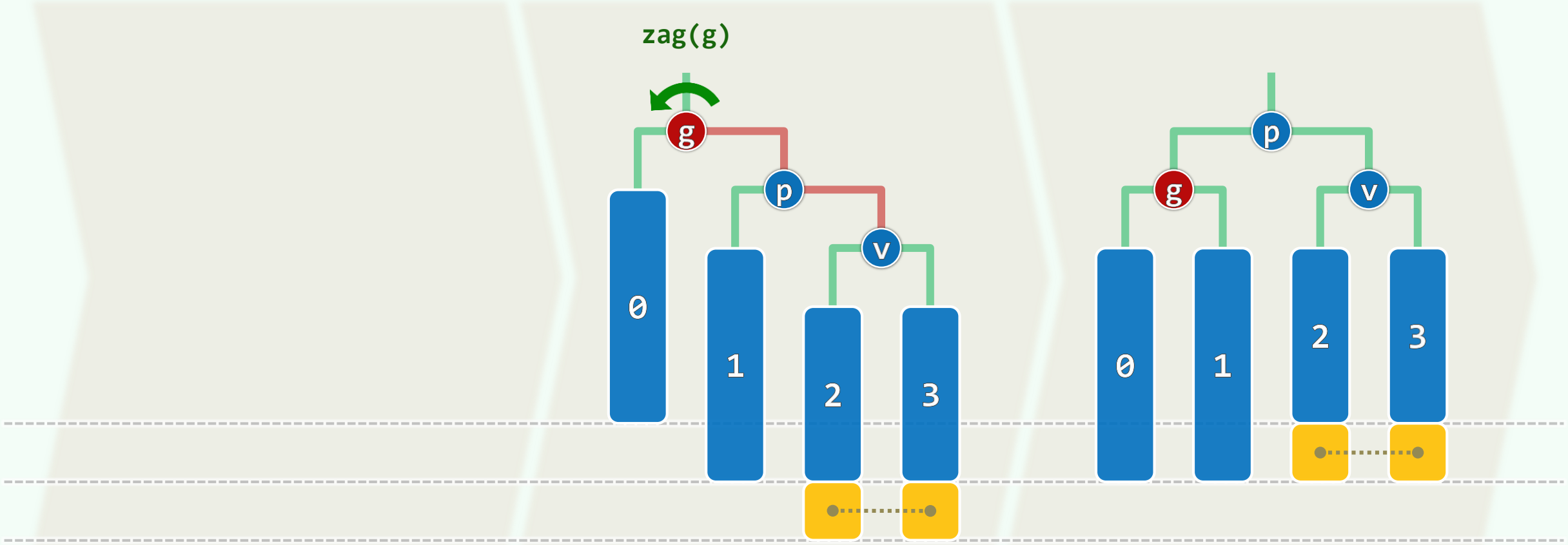
最低者g不低于x的祖父

❖ g经单旋调整后复衡

子树高度复原

❖ 更高祖先也必平衡

全树复衡



双旋

❖ 同时可有多个失衡节点

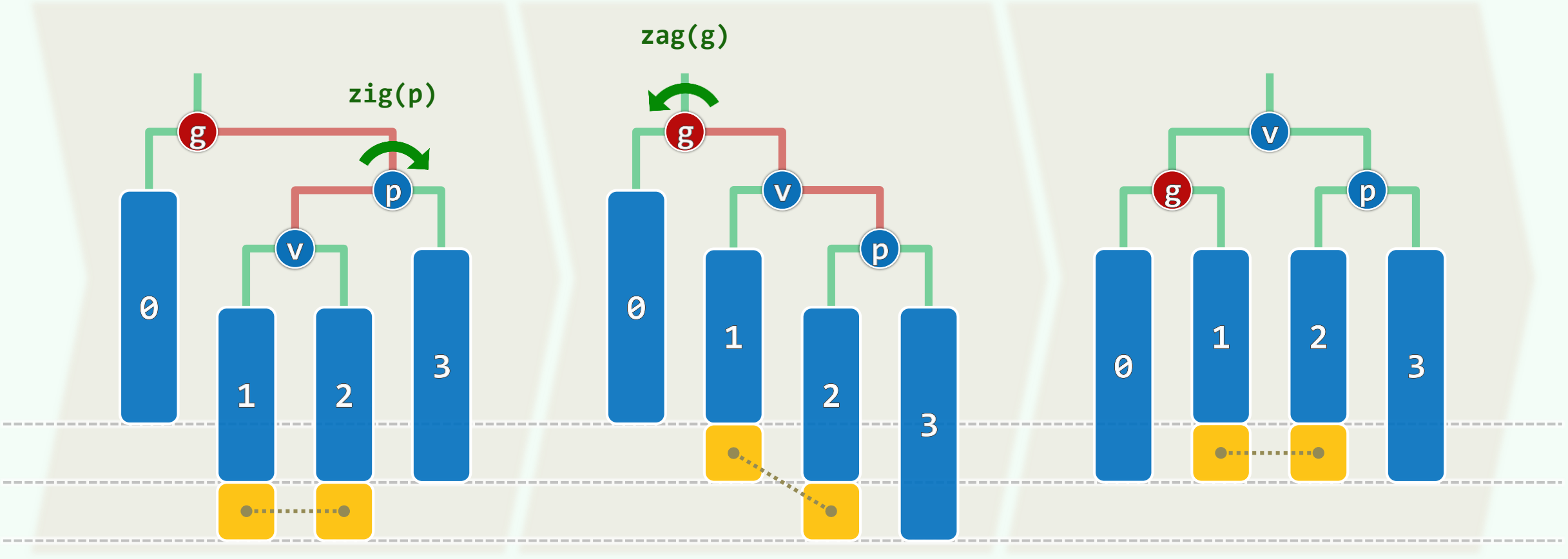
最低者g不低于x的祖父

❖ g经单旋调整后复衡

子树高度复原

❖ 更高祖先也必平衡

全树复衡



实现

```
❖ template <typename T> BinNodePosi<T> AVL<T>::insert( const T & e ) {
    BinNodePosi<T> & x = search( e ); if ( x ) return x; //若目标尚不存在
    BinNodePosi<T> xx = x = new BinNode<T>( e, _hot ); _size++; //则创建新节点
  // 此时，若x的父亲_hot增高，则祖父有可能失衡
    for ( BinNodePosi<T> g = _hot; g; g = g->parent ) //从_hot起，逐层检查各代祖先g
       if ( ! AvlBalanced( *g ) ) { //一旦发现g失衡，则通过调整恢复平衡
          FromParentTo(*g) = rotateAt( tallerChild( tallerChild( g ) ) );
          break; //局部子树复衡后，高度必然复原；其祖先亦必如此，故调整结束
       } else //否则（g仍平衡）
          updateHeight( g ); //只需更新其高度（注意：即便g未失衡，高度亦可能增加）
    return xx; //返回新节点位置
  }
```