06-B2

二叉搜索树

算法及实现：插入

邓俊辉

deng@tsinghua.edu.cn

算法

- 先借助search(e)

  确定插入位置及方向

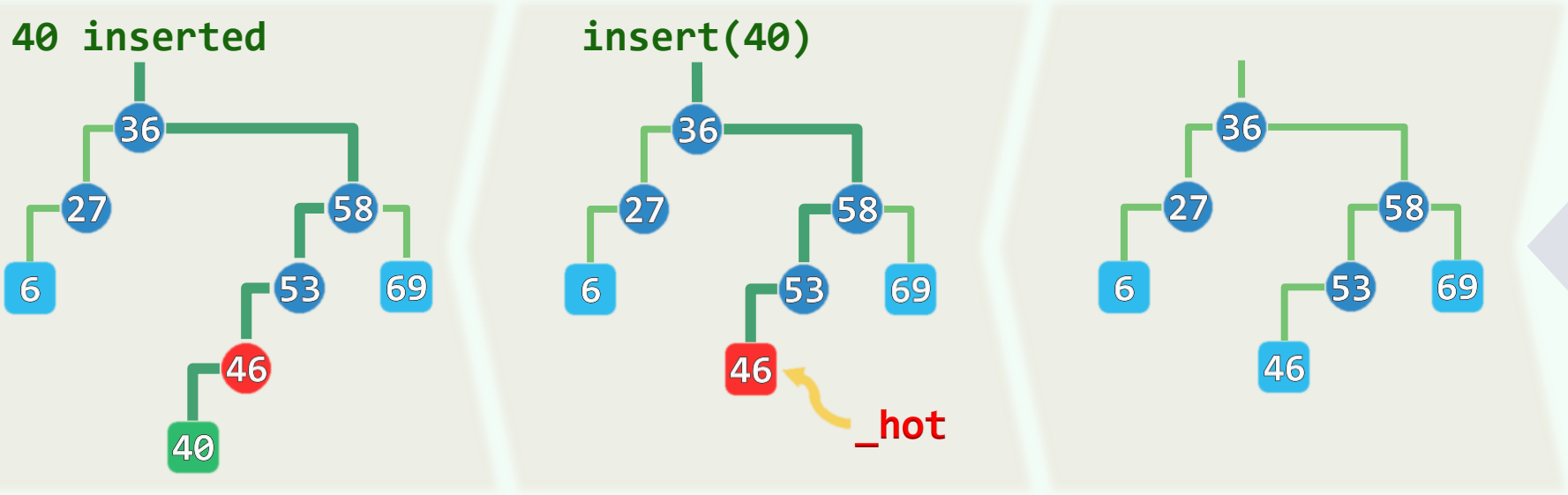
- 若e尚不存在，则再

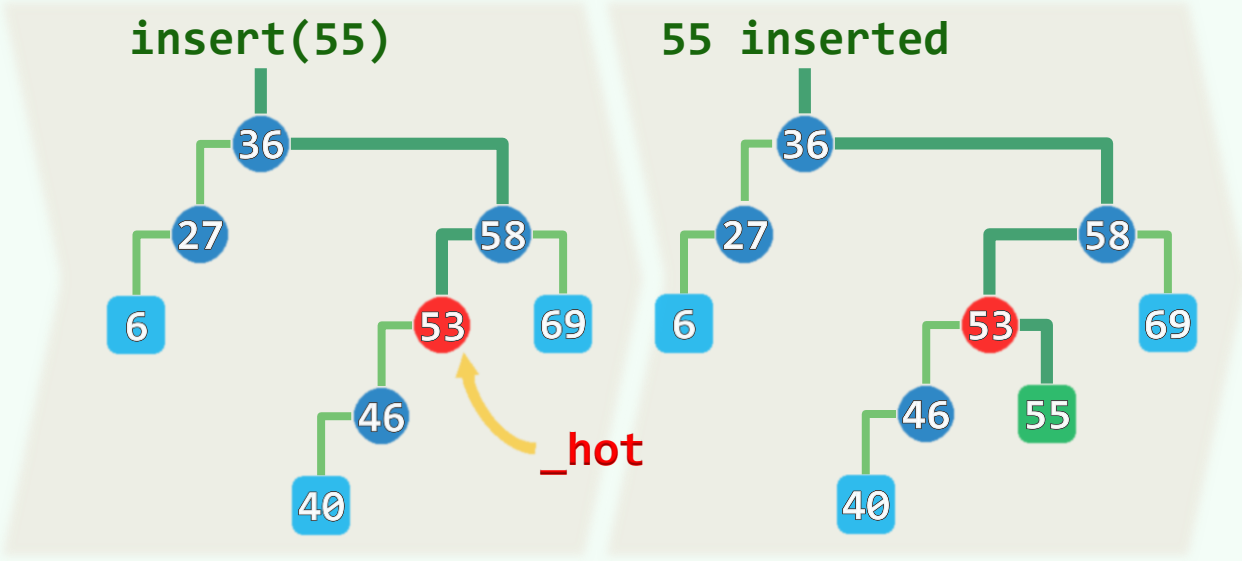
  将新节点作为叶子插入

  - _hot为新节点的父亲
  - v = search(e)

    为_hot对新孩子的引用

- 于是，只需令_hot通过v指向新节点





实现

```
template <typename T> BinNodePosi<T> BST<T>::insert( const T & e ) {
   BinNodePosi<T> & x = search( e ); //查找目标（留意_hot的设置）
   if ( ! x ) { //既禁止雷同元素，故仅在查找失败时才实施插入操作
      x = new BinNode<T>( e, _hot ); //在x处创建新节点，以_hot为父亲
      _size++; updateHeightAbove( x ); //更新全树规模，更新x及其历代祖先的高度
   }
   return x; //无论e是否存在于原树中，至此总有x->data == e
} //验证：对于首个节点插入之类的边界情况，均可正确处置
```

❖ **时间主要消耗于search(e)和updateHeightAbove(x)；均线性正比于x的深度，不超过树高**