ORF 201

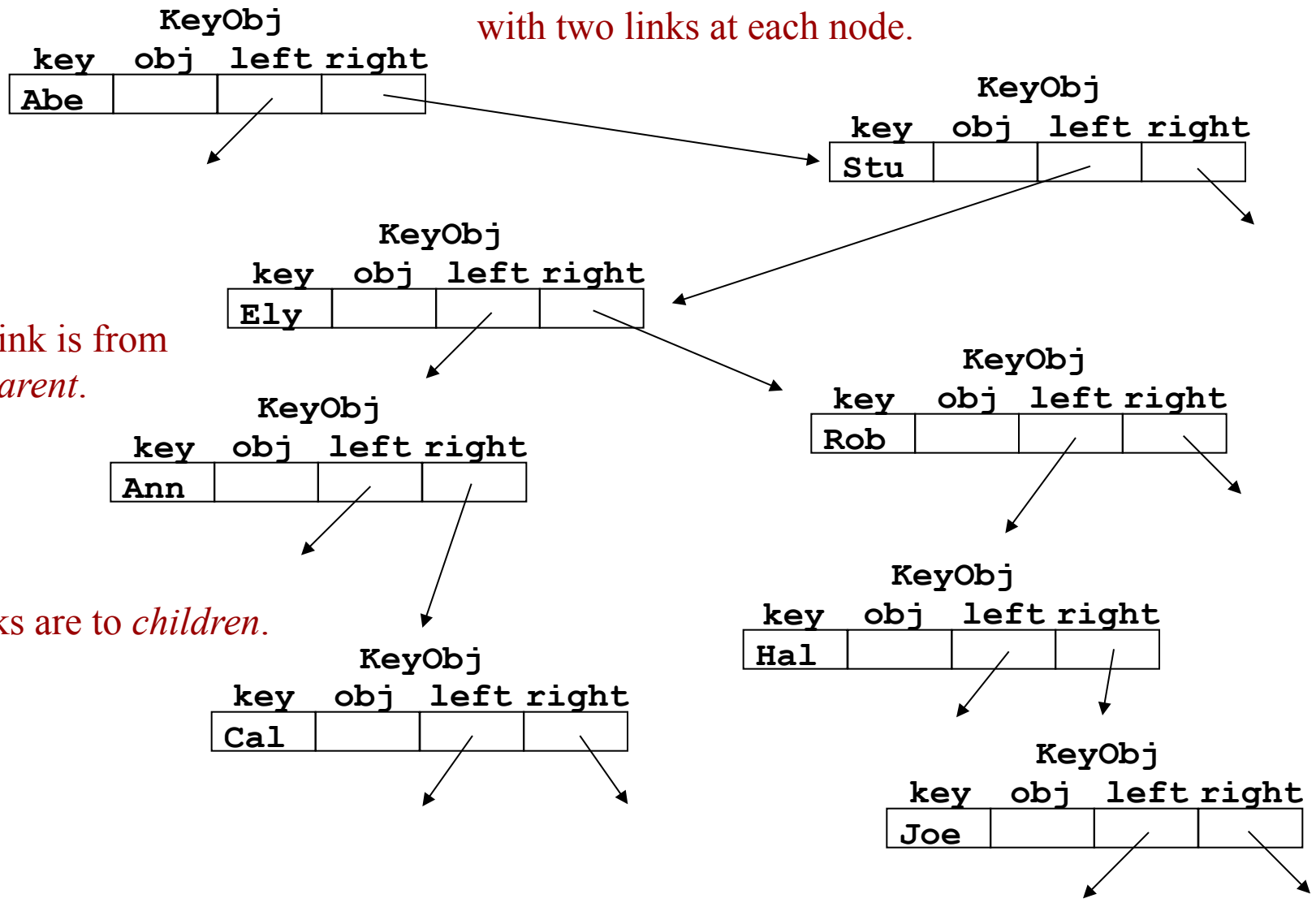
COMPUTER METHODS FOR PROBLEM SOLVING

Lecture 21

Binary Trees

Binary Tree

Similar to linked list but
with two links at each node.



A link is from
a *parent*.

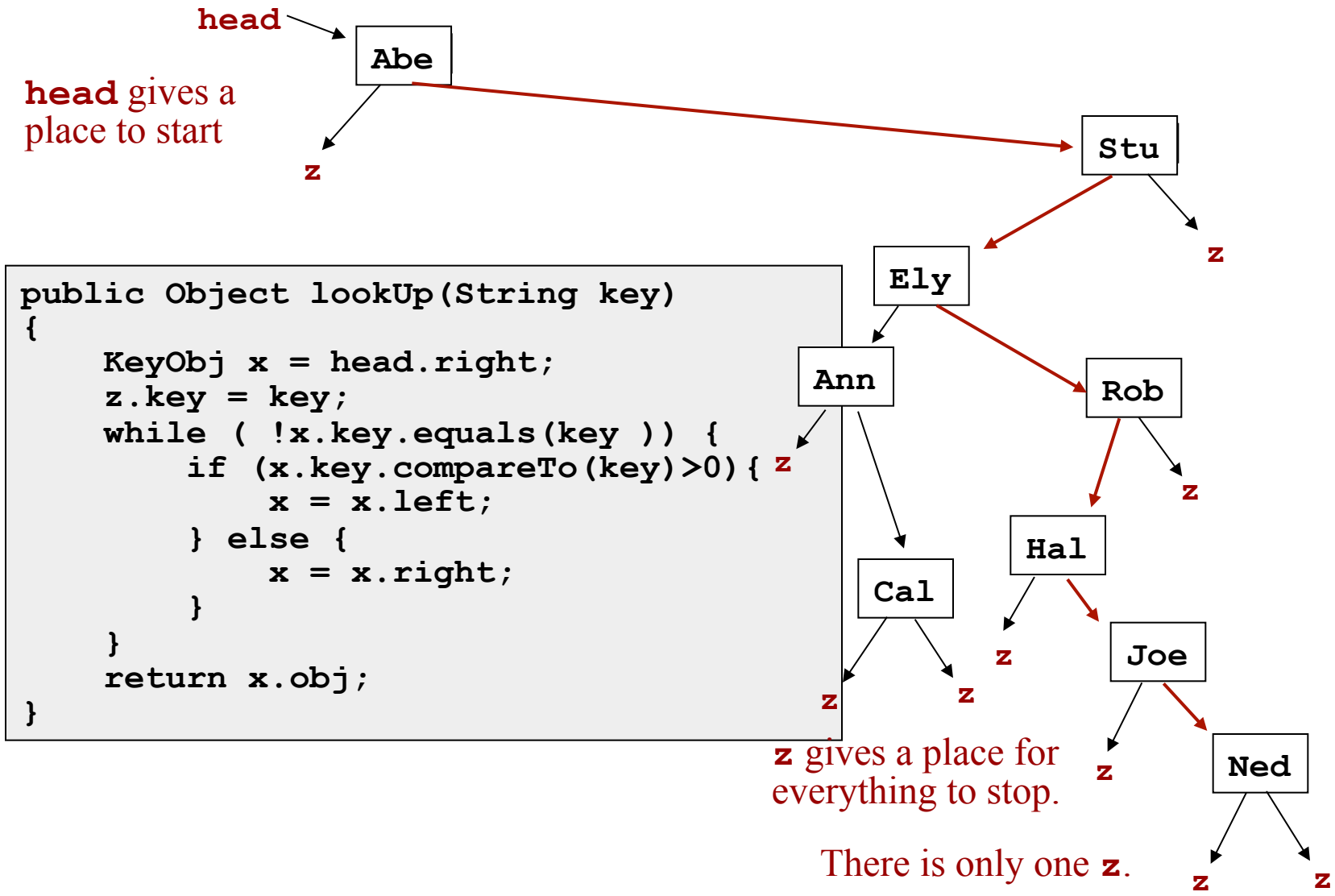
Links are to *children*.

Classes KeyObj and BTree

```
class KeyObj
{
     String key;
     Object obj;
     KeyObj left;
     KeyObj right;
}
```

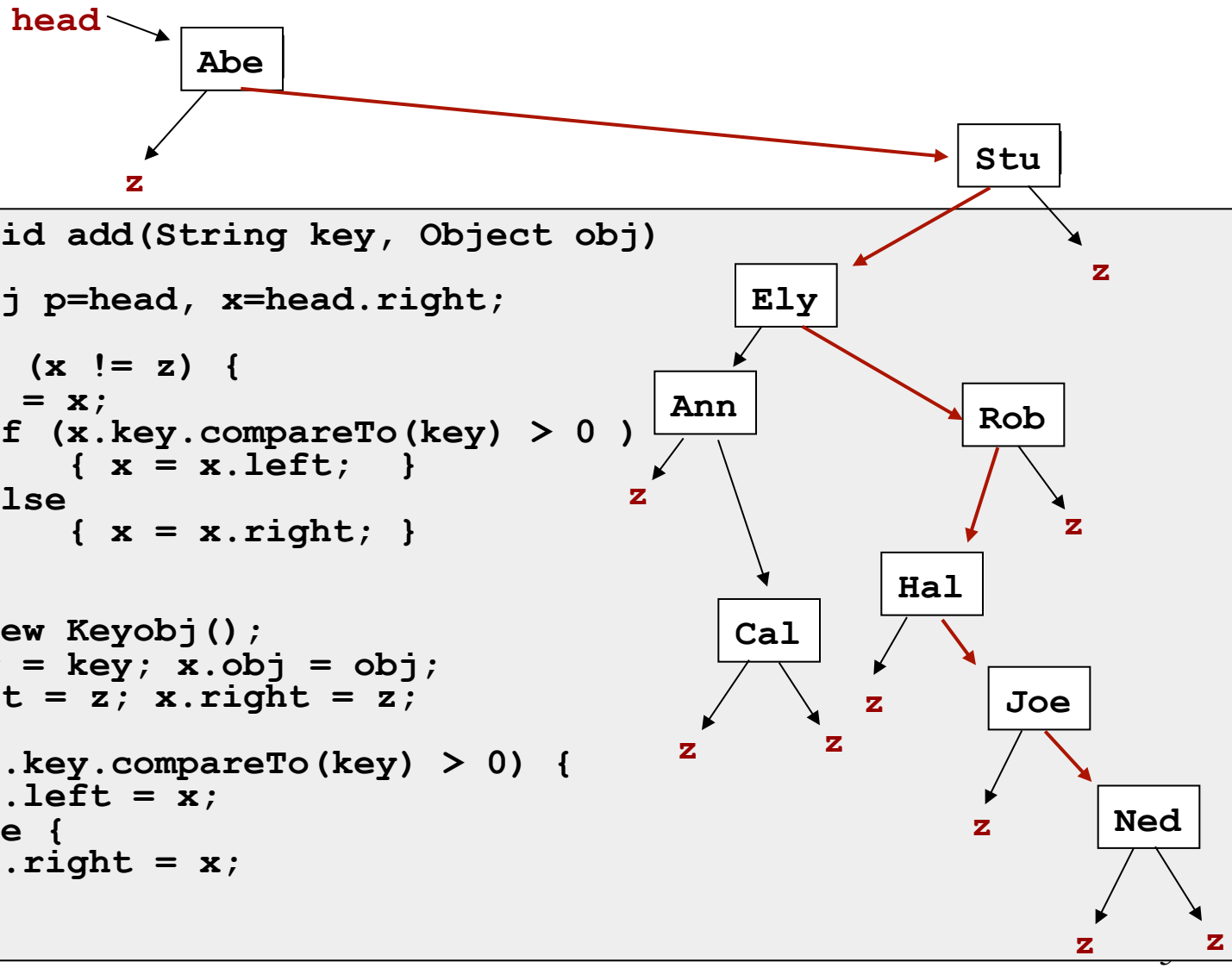
```
class BTree
{
     KeyObj z, head;
       .
       .
       .
}
```

Binary Tree - Search




```
public Object lookUp(String key)
{
    KeyObj x = head.right;
    z.key = key;
    while ( !x.key.equals(key )) {
        if (x.key.compareTo(key)>0){
            x = x.left;
        } else {
            x = x.right;
        }
    }
    return x.obj;
}
```

Binary Tree - Add



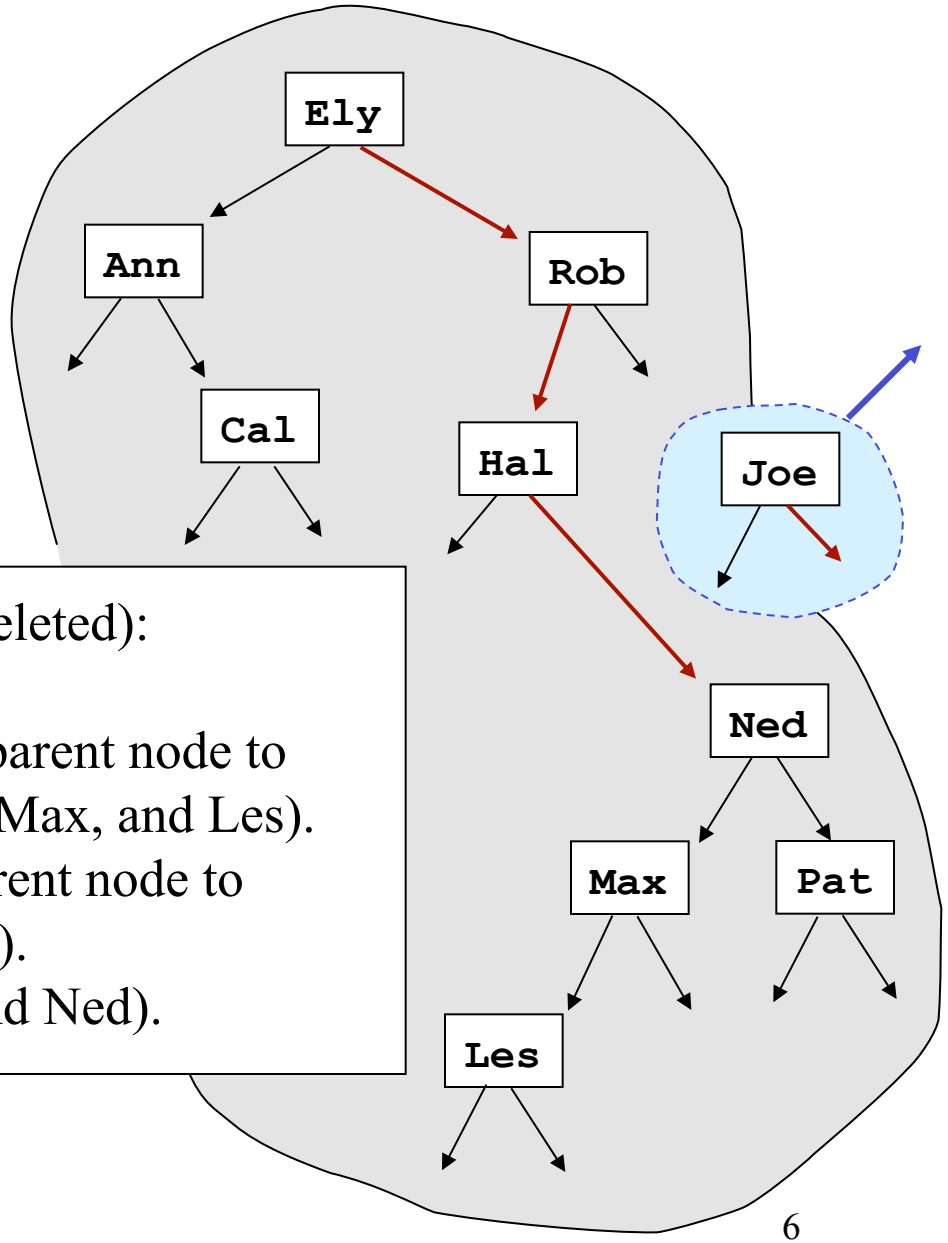

```
public void add(String key, Object obj)
{
    KeyObj p=head, x=head.right;

    while (x != z) {
        p = x;
        if (x.key.compareTo(key) > 0 )
            { x = x.left;  }
        else
            { x = x.right; }
    }

    x = new Keyobj();
    x.key = key; x.obj = obj;
    x.left = z; x.right = z;

    if (p.key.compareTo(key) > 0) {
        p.left = x;
    } else {
        p.right = x;
    }
}
```

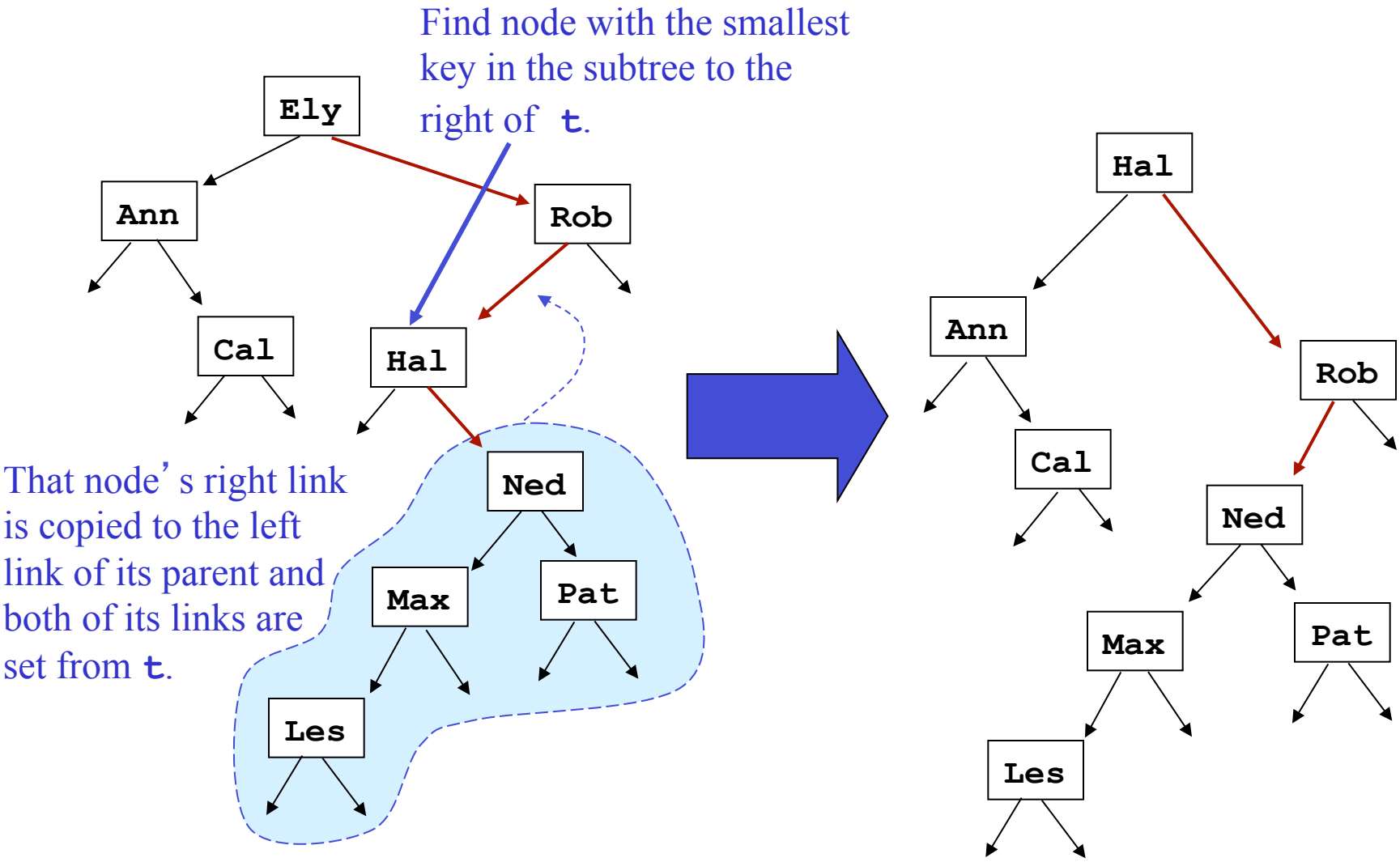
Binary Tree - Delete



Three cases (`t` denotes node to be deleted):

- `t` has no right child - set child of parent node to the left child of `t`. (Rob, Cal, Pat, Max, and Les).
- `t` has no left child - set child of parent node to the right child of `t` . (Ann and Hal).
- Otherwise - see next page. (Ely and Ned).

Deleting Ely



Delete - The Code

```
public void delete(String key)
{
    KeyObj c, p, x, t;
    z.key = key;
    p = head;
    x = head.right;
    while ( !x.key.equals(key )) {
        p = x;
        if ( x.key.compareTo(key) > 0) { x = x.left;  }
        else                           { x = x.right; }
    }
    t = x;
    if (t.right == z) {
        x = x.left;
    } else
    if (it.right.left == z) {
        x = x.right;
        x.left = t.left;
    } else {
        c = x.right;
        while (c.left.left != z) {
            c = c.left;
        }
        x = c.left;
        c.left = x.right;
        x.left = t.left;
        x.right = t.right;
    }
    if (p.key.compareTo(key) > 0) { p.left = x;  }
    else                          { p.right = x; }
}
```