算法导论 XII 章第 2 节习题

Louis1992
有需要可以联系我
我的 github欢迎大家帮我完成算法导论
我的博客

May 3, 2015

Exercises 12.2-1 Suppose that we have numbers between 1 and 1000 in a binary search tree and want to search for the number 363. Which of the following sequences could not be the sequence of nodes examined?

- 2, 252, 401, 398, 330, 344, 397, 363.
- 924, 220, 911, 244, 898, 258, 362, 363.
- 925, 202, 911, 240, 912, 245, 363.
- 2, 399, 387, 219, 266, 382, 381, 278, 363.
- 935, 278, 347, 621, 299, 392, 358, 363.

第 3 组不可能，$912 > 911$

Exercises 12.2-2 Write recursive versions of the TREE-MINIMUM and TREE-MAXIMUM procedures.

```
TREE-MINIMUM(x)
1  if left[x] ≠ NIL
2     then return TREE-MINIMUM(left[x])
3  return x
```

TREE-MAXIMUM(x)

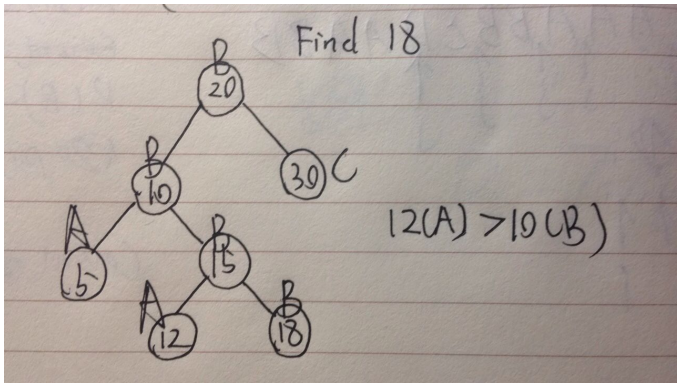
```
1  if right[x] ≠ NIL
2     then return TREE-MAXIMUM(right[x])
3  return x
```

Exercises 12.2-3 Write the TREE-PREDECESSOR procedure.

TREE-PREDECESSOR(x)

```
1  if left[x] ≠ NIL
2     then return TREE-MAXIMUM(left[x])
3  y ← p[x]
4  while y ≠ NIL and x = left[y]
5        do x ← y
6           y ← p[y]
7  return y
```

Exercises 12.2-4 Professor Bunyan thinks he has discovered a remarkable property of binary search trees. Suppose that the search for key k in a binary search tree ends up in a leaf. Consider three sets: A, the keys to the left of the search path; B, the keys on the search path; and C, the keys to the right of the search path. Professor Bunyan claims that any three keys a∈A, b∈B, and c∈C must satisfy a ≤ b ≤ c. Give a smallest possible counterexample to the professor's claim.



Exercises 12.2-5 Show that if a node in a binary search tree has two children, then its successor has no left child and its predecessor has no right child.

如果后继有左子女，那么那个左子女就是后继，所以后继不可能有左子女。同理前继没有右子女。

Exercises 12.2-6 Consider a binary search tree T whose keys are distinct. Show that if the right subtree of a node x in T is empty and x has a successor y, then y is the lowest ancestor of x whose left child is also an ancestor of x. (Recall that every node is its own ancestor.)

就是求后继的算法嘛

Exercise 12.2-7 inorder tree walk of an n-node binary search tree can be implemented by finding the minimum element in the tree with TREE-MINIMUM and then making $n-1$ calls to TREE- SUCCESSOR. Prove that this algorithm runs in Θ(n)time.

这个算法只遍历了每条边各两次，所以是 O(n) 的。

Exercise 12.2-8 Prove that no matter what node we start at in a height-h binary search tree, k successive calls to TREE-SUCCESSOR take $O(k+h)$ time.

上一题是这题的一个实例。只需要证明访问边的次数 $N \leq (4h+2k)$，假设 S 是开始起点，E 是结束点。

- 当 S 和 E 在同一条路径时（S 是 E 的祖先或者 E 是 S 的祖先），结论很明显。
- 当 S 和 E 不在同一条路径时，令 A 为 S 和 E 的最小公共祖先。考虑节点 S 到 A，后继函数回溯的成本至多只有 2h（思考的时候千万不要将后继到的点的成本算进去，那部分在 2k 里），A 到 E 的多余的成本至多也只有 2h, 其他都是正常的回溯到的点访问边 2k.

Exercise 12.2-9 Let T be a binary search tree whose keys are distinct, let x be a leaf node, and let y be its parent. Show that $key[y]$ is either the smallest key in T larger than $key[x]$ or the largest key in T smaller than $key[x]$.

若 x 是 y 的左叶子节点，那么 x 的后继是 y；若 x 是 y 的右叶子节点，那么 y 的后继是 x.