第 15 讲: 红黑树

姓名: 林凡琪　　**学号:** 211240042

评分: ______　　**评阅:** ______

2022 年 6 月 27 日

请独立完成作业，不得抄袭。
若得到他人帮助, 请致谢。
若参考了其它资料，请给出引用。
鼓励讨论，但需独立书写解题过程。

1　作业 (必做部分)

题目 1 (TC 18.1-5)

解答:
在将每个红色节点吸收到其黑色父节点之后，每个黑色节点可能包含 1 2（1 红色子节点）或 3（2 红色子节点）键，并且结果树的所有叶子都具有相同的深度，根据到红黑树的性质 5（对于每个节点，从节点到后代叶子的所有路径都包含相同数量的黑色节点)。因此，红黑树会变成度数 $t = 2$ 的 Btree，即 2-3-4 树。

题目 2 (TC 13.1-5)

解答:
是的。
性质一很容易满足, 因为之更改了一个节点, 并且没有更改为第三种颜色;
性质三也很容易满足, 因为没有引入新的叶子节点;
性质四被满足, 因为没有引入红色节点, 对这条性质没有影响;
性质五被满足, 因为我们将更改黑色节点数量的唯一路径是来自根的路径. 所有的路径上的黑色节点都将增加 1, 因此是平等的.

题目 3 (TC 13.1-7)

解答:
可能的 degree 是 0 到 5，这取决于黑色节点是否是根以及它是有一个还是两个红色子节点。深度最多可以缩小 1/2 倍。

题目 4 (TC 13.3-1)

解答:
如果我们选择将 z 的颜色设置为黑色，那么我们将违反作为红黑树的性质 5。因为从根到 z 下的叶子的任何路径都会比到其他叶子的路径多一个黑色节点。

题目 5 (TC 13.3-5)

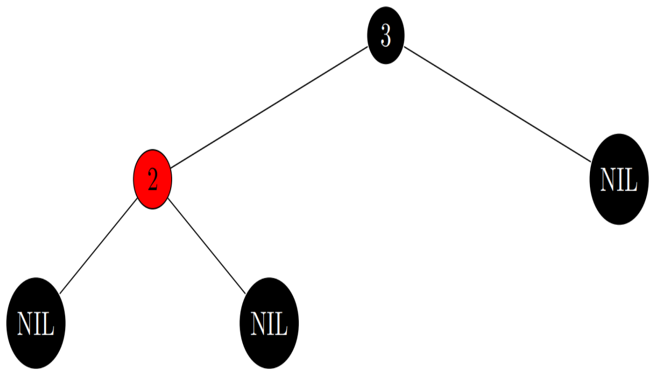
解答:
情形 1：z 和 $z.p.p$ 是 RED，如果循环终止，那么 z 不可能是根，因此修复后 z 是 RED .
情形 2：z 和 $z.p$ 是 RED, 旋转后 $z.p$ 不可能是根，因此修复后 $z.p$ 是 RED。
情形 3：z 是 RED 并且 z 不能是根，因此修复后 z 是 RED。因此，总是至少有一个红色节点。

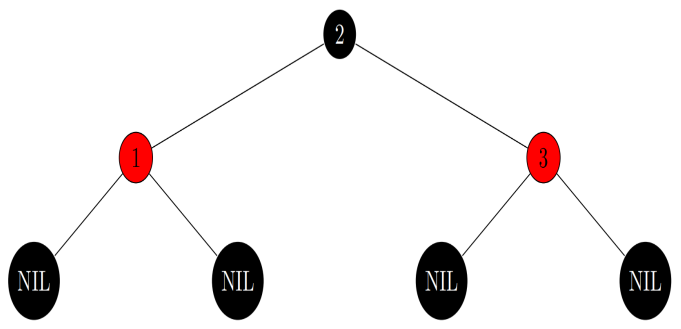
题目 6 (TC 13.4-1)

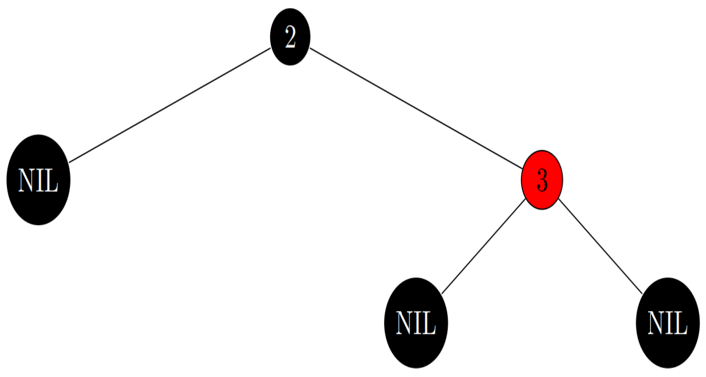
解答:
情形 1：转换为 2、3、4。情形 2：如果终止，则子树的根（新的 x）设置为黑色。情形 3：转换为 4。情形 4：根（即新的 x）设置为黑色。

题目 7 (TC 13.4-7)

解答:
不一定和原来一样.
比如:
原红黑树树如图:
　　插入 1 后:　　删除 1 后:　　显然, 最后和刚开始不同.







2 作业 (选做部分)

题目 1 (TC Problem 13-3: AVL trees)

解答:

---

3 Open Topics

4 订正

题目 1 (D_n1-13.2)
请证明: D_n (定义见阅读材料 Example 15.1 (c)) 是 Boolean algebra 当且仅当 $n = p_1p_2\cdots p_k$ (for some k), 这里 p_i 皆为素数且互异。

解答:
Suppose, to the contrary. There exists Boolean algebra $D_n = p_1^{t_1}p_2^{t_2}\cdots p_k^{t_k}$(for some k, and $t_i \in \mathbb{N}^+$) satisfied these p_i are all different prime numbers but t_i are not all 1.
Assume that $t_1 \neq 1$.
Let $a = p_1$, and $a' = p_1^{t_1-1}p_2^{t_2}\cdots p_k^{t_k}$.
So we can have that $a * a' = gcd(a, a') = p_1$
With the definition, $a * a' = 1$ but $p_1 \neq 1$, which 1 is the zero element.
It is contradict with the definition of Boolean algebra.
So if D_n is Boolean algebra, then $n = p_1p_2\cdots p_k$ (for some k), and these p_i are all different prime numbers. (1)

If D_n satisfied $n = p_1p_2\cdots p_k$ (for some k), and these p_i are all different prime numbers, then D_n is a Boolean algebra as the textbook SM explained. (2)
(1) $\wedge$ (2) $\rightarrow$ D_n is Boolean algebra if and only if $n = p_1p_2\cdots p_k$ (for some k), and these p_i are all different prime numbers.

题目 2 (2-1.5 DH Problem 5.14 (*a*, *b*): *Pal4*(*S*))
Note: You don't have to consider *Pal3* in Problem 5.13.
(a)Construct a correct solution to the problem of checking efficiently whether a string is a palindrome, following the general ideas of Pal2 and Pal3. Call your algorithm Pal4.
(b) Prove the total correctness of Pal4.

解答:
(a)

Algorithm 1 judge whether the string is a palindrome

```
procedure Pal4($S$)
    $X \leftarrow S$
    $E \leftarrow true$
    while $X \neq \mathbf{\Lambda}$ and $E == true$ do
        if eq(head($X$),last($X$)) then
            $X \leftarrow$ all-but-last(tail(X))
        else
            $E \leftarrow false$
        end if
    end while
    return E
end procedure
```

(b)
The partially correction has been proven in Problem4(a).
Now we prove the algorithm will terminate.
If the string is not a a palindrome, then line 8 will make E=false in one turn, and the while-do loop will break.
If the string is a palindrome, then every time the while-do loop is traversed, X is made shorter by at most two symbol, and finally becomes $\mathbf{\Lambda}$. The while-do loop will break.
Therefore, if the input string is legal, the algorithm will terminate.
Since Pal4 is partially correct and will terminate finally, Pal4 is totally correct.

题目 3 (2-2.4 TC Exercise 3.1-6)
Prove that the running time of an algorithm is $\Theta(g(n))$ if and only if its worst-case running time is $O(g(n))$ and its best-case running time is $\Omega(g(n))$

解答:
Assume that the running time of the algorithm is $f(n)$.
$f(n) = \Theta(g(n))$

$\leftrightarrow \exists n_0, c_1, c_2, (\forall n, (n \geq n_0 \rightarrow 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)))$
$\leftrightarrow \exists n_0, c_1, c_2, (\forall n, (n \geq n_0 \rightarrow (0 \leq c_1 g(n) \leq f(n)) \wedge (0 \leq f(n) \leq c_2 g(n))))$
$\leftrightarrow f(n) = O(g(n)) \wedge f(n) = \Omega(g(n))$
The $O(g(n))$ describes the upper bound of $f(n)$, and $\Omega(g(n))$ describes the lower bound of $f(n)$.
Therefore, we can have that the running time of an algorithm is $\Theta(g(n))$ if and only if its worst-case running time is $O(g(n))$ and its best-case running time is $\Omega(g(n))$.

题目 4 (2-4.4 TC 4.5-4)
Can the master method be applied to the recurrence $T(n) = 4T(n/2) + n^2 \lg n$? Why or why not? Give an asymptotic upper bound for this recurrence.

解答:
The master method can not be used to solve that, since $\frac{n^2 \lg n}{n^{\log_2 4}} = \lg n$ which is not not polynomially larger.
On the recursion-tree, The depth is $log_2 n + 1$, and the total cost is $n^2 log_2 n - n^2 i$ nodes in the i^{th} depth.
$\sum_{i=0}^{\log_2 n} n^2 log_2 n - n^2 i = O(\sum_{i=0}^{\log_2 n} n^2 log_2 n)$
Therefore, $T(n) = O(n^2 log^2 n)$.

题目 5 (2-6.1 DH 4-8)
Prove that the maximal distance between any two points on a polygon occurs between two of the vertices.

解答:
对于在凸包上的任意两点，我们一定可以分别过这两个点做直线，满足两条直线平行，且直线间距离等于两点间距离。
将这两条直线分别向外平移。对于每条直线，当他仅经过凸包上的一个顶点，或恰好包含凸包上的一条边时停止平移。
此时显然有平移后直线间距离大于等于原来选取的两点间距离。
此时分为 3 种情况：
1, 若平移后两条直线都恰好经过凸包的一个顶点，显然有两顶点距离大于等于直线间距离。因此两顶点距离大于等于原选点距离。得证。
2, 若平移后恰好有一条直线包含凸包上的一条边，显然有另一条直线所经过的顶点，到这条边两个顶点的距离最大值，大于等于原选点距离。得证。
3, 若平移后两条直线都包含凸包上的边，显然有这两条边的四个顶点间的距离最大值，大于等于原选点距离。得证。
综上，在凸包中选取任意两点，都能找到凸包上的两个顶点，满足顶点间距离大于等于所选点。
原命题得证。

题目 6 (2-6.3 DH 4-12)
Write high-level pseudocode of the greedy algorithm described in the text for finding a minimal spanning tree.

Algorithm 2 minimal spanning tree

解答:

```
procedure COST(A[],n)                    ▷ A 为一组有连边关系的点的二元组,n 为组数
    $ans \to 0$
    sort(A)                              ▷ 将二元组按花费从小到大的顺序排序
    for $i \leftarrow 1$ to $n$ do
        if is_connected(A[i].x,A[i].y)==False then
            connext A[i].x and A[i].y
            $ans \to ans + A[i].cost$
        end if
    end for
    return ans
end procedure
```

题目 7 (2-15.2 TC 13.1-7)

Describe a red-black tree on n keys that realizes the largest possible ratio of red internal nodes to black internal nodes. What is this ratio? What tree has the smallest possible ratio, and what is the ratio?

解答:

(1)

当红黑树中满足：每个黑色节点的子节点若不为 nil，则为红节点时，其内节点的红黑比最大。

若 $n = 2^{2k} - 1$，$k \in \mathbb{N}^+$ 时，红黑树全黑节点层与全红节点层交替出现，此时最大红黑比为 2

(2)

当红黑树中所有内节点都为黑色节点时，其红黑比最小，为 0.

题目 8 (2-15.3 TC 13.3-1)

In line 16 of RB -INSERT, we set the color of the newly inserted node z to red. Observe that if we had chosen to set z 's color to black, then property 4 of a redblack tree would not be violated. Why didn't we choose to set z 's color to black?

解答:

将节点设为红色，会破坏性质 2/4，调整这一后果影响的节点较为局限，便于操作

若将节点设为黑色，性质 5 则会被破坏。若从维护性质 5 的角度去解决，问题会变得更加棘手

---

5 反馈