算法与复杂性　作业十二

516021910528 - SHEN Jiamin

2020 年 5 月 7 日

1　0430

1. 已知 n 个矩形，这些矩形的边都平行于坐标轴

   (a) 求出这些矩形的交集

   > 求两个矩形的交集所需时间为 $O(1)$。任取两个求交集，求得的交集再与下一个矩形求交集即可。该交集若存在则仍然是一个矩形。
   >
   > **伪代码见算法1**

算法 1 矩形交集

输入: 一组矩形 $R = \{\langle y_u, x_l, y_b, x_r \rangle\}$ ▷ 矩形的四条边，按上左下右的顺序

输出: 这些矩形的交集 $\langle Y_u, X_l, Y_b, X_r \rangle$ ▷ 若不存在交集，则这四条边不构成矩形

$\langle Y_u, X_l, Y_b, X_r \rangle \leftarrow \textsc{Select}(R)$ ▷ 从集合中任选一个矩形作为当前的交集

for $\langle y_u, x_l, y_b, x_r \rangle$ in R **do**

　　$Y_u \leftarrow \min(Y_u, y_u), Y_d \leftarrow \max(Y_d, y_d)$

　　$X_l \leftarrow \max(X_l, x_l), X_r \leftarrow \min(X_r, x_r)$

　　if $Y_u \leq Y_d \vee X_l \geq X_r$ **then**

　　　　break

　　end if

end for

   (b) 求出这些矩形能够覆盖的面积

   > 使用扫描线算法。
   >
   > 事件列表为所有矩形的竖直边。扫描线状态为当前扫描线穿过的所有矩形。
   >
   > 当扫描线状态发生变化时，计算当前事件点到上一个事件点之间的有效面积，然后更新当前扫描线上的有效高度。
   >
   > **伪代码见算法2**

算法 2 矩形并集面积

输入: 一组矩形 $R = \{\langle y_u, x_l, y_b, x_r \rangle\}$ ▷ 矩形的四条边，按上左下右的顺序

输出: 这些矩形的覆盖面积

对所有矩形按两条竖直边的横坐标建最小化堆 $events$，有元素 $2n$ 个 ▷ $O(n)$
$state \leftarrow$ RedBlackTree.New
$xlast \leftarrow \min(x), ylast \leftarrow 0$
$space \leftarrow 0$
for e in $events$ **do**
  $x, y_u, y_d \leftarrow$ Extract(e)
  $space \leftarrow space + (x - xlast) \times ylast$
  if x_l in e **then** ▷ 遇到左边，出现一个新的矩形
    RBTree.Insert($state$, key $=y_u$, value $=(y_u$, up$)$)
    RBTree.Insert($state$, key $=y_b$, value $=(y_b$, bottom$)$)
  else if x_r in e **then** ▷ 遇到右边，这个矩形消失
    RBTree.Remove($state$, key $=y_u$)
    RBTree.Remove($state$, key $=y_b$)
  end if
  $xlast \leftarrow x, ylast \leftarrow 0$ ▷ 更新当前的有效高度
  $current \leftarrow$ Stack.New
  for $(y, type)$ in $state$ **do**
    if $type =$ up **then**
      Stack.Push($current, y$)
    else if $type =$ bottom **then**
      $ytop \leftarrow$ Stack.Pop($current$)
      if Stack.empty **then**
        $ylast \leftarrow ylast + (y - ytop)$
      end if
    end if
  end for
end for

2. 求 $n!$ 包含质因子 p 的数量，例如 6 含有 4 个 2，2 个 3 和 1 个 5。并给出算法的时间复杂性

伪代码见算法3

时间复杂度为 $O(n)$

算法 3 质因子数量

输入: 正整数 n，素数 p

输出: $n!$ 包含质因子 p 的数量 C

$cnt[1 \ldots n] \leftarrow \{0\}$
$C \leftarrow 0$
for $k \leftarrow p$ up to n **do**
    if $k \equiv 0 \bmod p$ **then**
        $cnt[k] \leftarrow cnt[k/p] + 1$
        $C \leftarrow C + cnt[k]$
    end if
end for

3. 设 Fibonacci 数列的定义为:

$$F(n) = \begin{cases} 1 & ,n = 1,2 \\ F(n-1) + F(n-2) & ,n > 2 \end{cases}$$

证明每个大于 2 的整数 n 都可以写成至多 $\log n$ 个 Fibonacci 数之和，并设计算法对于给定的 n 寻找这样的表示方式

首先设计用若干 Fibonacci 数之和表示给定大于 2 的整数 n 的算法，并由此证明每个大于 2 的整数 n 都可以写成 Fibonacci 数之和。然后证明在这样的表示中，使用的 Fibonacci 数至多有 $\log_2 n$ 个。

对于一个大于 2 的整数 n，若 n 是 Fibonacci 数，则命题显然成立。若 n 不是 Fibonacci 数，则一定存在一个 $m(m > 2)$，使得

$$F(m) < n < F(m+1)$$

又

$$F(m+1) = F(m) + F(m-1)$$

所以

$$0 < n - F(m) < F(m-1)$$

不妨设 $n' = n - F(m)$。若 n' 是 Fibonacci 数，则 $n = F(m) + n'$ 表示成了两个 Fibonacci 数之和。否则若 $n' > 2$，则令 $n = n'$，重复上述步骤，直到 $0 < n' \leq 2$。此时 $n' = 1$ 或 2 仍然是 Fibonacci 数。

所以每个大于 2 的整数 n 都可以用这种算法写成若干个 Fibonacci 数之和。

伪代码见算法4

注意到

$$F(m') < n' < F(m'+1) \le F(m-1) < F(m) < n < F(m-1)$$

所以每个 Fibonacci 数至多使用一次，且不会有连续两个 Fibonacci 出现在结果中。

又不超过 n 的最大 Fibonacci 数为 $F(m-2)$，即

$$\frac{F(m)}{F(m')} \ge \frac{F(m)}{F(m-2)} = \frac{F(m-1)+F(m-2)}{F(m-2)} = 2 + \frac{F(m-3)}{F(m-2)} \ge 2 \quad (m \ge 3)$$

即结果中相邻两个 Fibonacci 数之间的倍数一定超过 2。所以一定不超过 $\log_2 n$ 个。

算法 4 Fibonacci 表示

输入: 正整数 n

输出: 若干个 Fibonacci 数 $repr$，其和为 n

```
repr ← Vector.New
Fib ← Stack.New
Stack.Push(Fib, 1)
last ← 1
repeat
    next ← last + Stack.top(Fib)
    Stack.Push(Fib, last)
    last ← next
until last > n
repeat
    next ← Stack.Pop(Fib)
    if n ≥ next then
        Vector.PushBack(repr, next)
        n ← n − next
    end if
until n = 0
```

2　0507

4. 设有复数 $x = a + b\boldsymbol{i}$ 和 $y = c + d\boldsymbol{i}$，设计算法，只用 3 次乘法计算乘积 xy

$$xy = (a + b\boldsymbol{i})(c + d\boldsymbol{i}) = (ac - bd) + (ad + cb)\boldsymbol{i}$$

1. $A = ad$
2. $B = bc$
3. $C = (a + b)(c - d) = ac - ad + bc - bd$

$$\begin{aligned} ac - bd &= C + A - B \\ ad + cb &= A + B \end{aligned}$$

5. 设 P 是一个 n 位十进制正整数。如果将 P 划分为 k 段，则可得到 k 个正整数，这 k 个正整数的乘积称为 P 的一个 k 乘积。

   (a) 求出 1234 的所有 2 乘积

$$\begin{aligned} 1 \times 234 &= 234 \\ 12 \times 34 &= 408 \\ 123 \times 4 &= 492 \end{aligned}$$

   (b) 对于给定的 P 和 k，求出 P 的最大 k 乘积的值

使用动态规划求解。设 $Num(n, i)$ 是整数 n 的最低 i 位构成的数。设 $Prod(n, k)$ 是整数 n 的最大 k 乘积。则

$$Prod(n, k) = \begin{cases} n & , k = 1 \\ Num(n, i) \cdot \max_i Prod(Num(n, i), k - 1) & , k > 1 \end{cases}$$

6. 分析在一般微机上如何计算二项式系数 C_n^k

输入: 正整数 n, k
输出: C_n^k

$result \leftarrow 1$
$k \leftarrow \min(k, n-k)$
for $i \leftarrow 1$ to k **do**
    $result \leftarrow result \times (n - i + 1)$
    $result \leftarrow result \div i$
end for
return $result$