1819. 序列中不同最大公约数数目解析

张翼翔*

2023/1/14

1 方法一: 枚举

1.1 思路与算法

题目要求找到所有非空子序列中不同的最大公约数的数目, 我们可以尝试枚举所有可能的最大公约数. 假设 p 为一个序列 $A=[a_0,a_1,\cdots,a_k]$ 的最大公约数, 令 $a_i=ci\times p$, 则序列即为 $A=[c_0\times p,c_1\times p,c_2\times p,\cdots,c_k timesp]$, 根据最大公约数的性质可知此时 $gcd(a_0,a_1,a_2,\cdots,a_k)=p$, 则可以推出 $gcd(c_0,c_1,c_2,\cdots,c_k)=1$. 此时我们在序列 A 中添加 p 的任意倍数 $a_{k+1}=c_{k+1}\times p$ 时, 则序列 A 的最大公约数依然为 p, 即此时 $gcd(a_0,a_1,a_2,\cdots,a_k,a_{k+1})=p$.

根据以上推论我们可以得出结论, 如果 x 为数组 $nums$ 中的某个序列的最大公约数, 则数组中所有能够被 x 整除的元素构成的最大公约数一定为 x. 这样的数我们也称之为 **基本的**. 存在以 x 为最大公约数的充分必要条件就是:

$$\text{对于}\forall m\forall x((m\in[1,\max(nums)]\to m\equiv 0(\bmod\ x))\to(m\geqslant y)).\qquad(1)$$

*E-mail:21371055@buaa.edu.cn

根据上述结论, 我们可以枚举所有可能的最大公约数 x, 其中 $x \in [1, \max(nums)]$, 然后对数组中所有可以整除 x 的元素求最大公约数, 判断最后求出的最大公约数是否等于 x 即可. 如果等于, 说明这些数恰好以 x 为最大公约数, 则 $ans++$. 否则还有比 x 更大的公约数 y, x 不是最大公约数.

1.2 代码

```
class Solution {
  public:
      int countDifferentSubsequenceGCDs(vector<int>& nums) {
          int maxVal = 0;
          int ans = 0;
          for (vector<int>::iterator it
        = nums.begin(); it != nums.end(); ++it)
          {
              maxVal = max(maxVal, (*it));
          }
          vector<bool> occured(maxVal + 2, false);
          for (vector<int>::iterator it
          = nums.begin(); it != nums.end(); ++it)
          {
              occured[(*it)] = true;
          }
          for (int i = 1; i <= maxVal; i++)
          {
              int gcdVal = 0;
              for (int k = i; k <=
              maxVal; k = k + i)
              {
```

```
                    if (occured[k])
                    {
                        if (gcdVal == 0)
                        {
                            gcdVal = k;
                        }
                        else
                        {
                            gcdVal = __gcd(gcdVal, k);
                        }
                    }
                }
                ans += (i == gcdVal);
            }
            return ans;
        }
        //O(n + maxVal log(maxVal))
    };
```

1.3 复杂度分析

1.3.1 时间复杂度

$O(n+\max(nums)\log(\max(nums)))$, 其中 n 表示数组的长度, $\max(nums)$ 表示数组中的最大元素. 我们首先需要遍历一遍数组, 然后从 1 到 $\max(nums)$ 一次枚举每个可能的最大公约数 (公式 (1)). 对于给定的数 x, 每次时间复杂度为 $\frac{\max(nums)}{x}$.

而 $n+\frac{n}{2}+\frac{n}{3}+\frac{n}{3}+\cdots+1=\sum_{i=1}^{n}\frac{n}{i}\approx n\log(n)$, 因此在枚举的时候需要的时间为 $\max(nums)\log(\max(nums))$, 总时间复杂度 $O(n+\max(nums)\log(\max(nums)))$.

1.3.2 空间复杂度

$O(\max(nums))$, 其中 $\max(nums)$ 表示数组中的最大元素. 我们需要一个 $O(\max(nums))$ 空间来标记数组中的每个元素是否出现过 ($[1, \max(nums)]$ 对应元素).