动态规划 && 网易合唱团笔试题

W.J.Z

1 问题描述

有 n 个学生站成一排，每个学生有一个能力值，牛牛想从这 n 个学生中按照顺序选取 k 名学生，要求相邻两个学生的位置编号的差不超过 d，使得这 k 个学生的能力值的乘积最大，你能返回最大的乘积吗？

每个输入包含 1 个测试用例。每个测试数据的第一行包含一个整数 n (1 <= n <= 50)，表示学生的个数，接下来的一行，包含 n 个整数，按顺序表示每个学生的能力值 ai（-50 <= ai <= 50）。接下来的一行包含两个整数，k 和 d (1 <= k <= 10, 1 <= d <= 50)。

2 动态规划

2.1 子问题分解

分解方式: 从 n 个学生中选取 k 名学生 » 从 n 个学生中选取 1 名学生；从 n 个学生中选取 2 名学生;...; 从 n 个学生中选取 k 名学生。

2.2 状态矩阵

两种分解方式的状态矩阵相同，状态总数为 $k * n$ 个，令 pMax(k,i) 代表该状态下 k 个能力值乘机最大的结果，k 代表从 n 个学生中选取 k 名学生且最后一名学生的标号为 i，num(i) 代表第 i 个学生的能力值.Pmax 矩阵与子问题分解对应起来很好理解，至于为什么 i 为选取学生的最后一名的标号将在状态转移方程中讲解。

表 1: pMax 矩阵

0	1	2	...	n
1	pMax(1,1)	pMax(1,2)	...	pMax(1,n)
2	pMax(2,1)	pMax(2,2)	...	pMax(2,n)
...	...	...	...	...
k	pMax(k,1)	pMax(k,2)	...	pMax(k,n)

2.3 状态转移方程

状态转移方式很好求解，根据状态矩阵在实际情况中运算几步既可以得到一些方程式，让后将方程式中的参数换位形参符号即可。

第 1 阶段：从 n 名学生中选取 1 名学生；显然每个学生都可能被单独抽中，即

$$pMax(1,1) = num(1), \ldots, pMax(1,n) = num(n) \tag{1}$$

第 2 阶段：从 n 名学生中选取 2 名学生；根据 pMax(k,i) 代表从 n 个学生中选取 k 名学生且最后一名的标号为 i 得

$$pMax(k,i) = 0, i < k \tag{2}$$

即 $pMax(2,1) = 0$, 表示此种方案不存在。

$$pMax(2,2) = max(pMax(1,1) * num(2), pMax(1,2) * num(2)) \tag{3}$$

公式 3就已经告诉我们状态转移公式的普遍形式了。题目中要求选取相邻两个学生之间编号之差不超过 d, 将该要求导入公式同时将公式中的实参换为形参得

$$pMax(k,i) = max(pMax(k-1,i-1)*num(i), pMax(k-1,i-2)*num(i), \ldots, pMax(k-1,i-d)*num(i)) \tag{4}$$

特殊情况：公式 4可以很好的应用在能力值都为正值的情况下，但实际情况能力值还可能存在负值，公式一旦遇到能力值为负值的学生就会跳过，但有时负负得正的取值可能比正正得正的取值还要大，因此还需要一个 $pMin(k,i)$ 矩阵来考虑负值的情况。

表 2: pMin 矩阵

0	1	2	...	n
1	pMin(1,1)	pMin(1,2)	...	pMin(1,n)
2	pMin(2,1)	pMin(2,2)	...	pMin(2,n)
...	...	...	...	...
k	pMin(k,1)	pMin(k,2)	...	pMin(k,n)

公式 4要改写为

$$\begin{aligned} pMax(k,i) = max(&pMax(k-1,i-1) * num(i), pMin(k-1,i-1) * num(i), \ldots, \\ &pMax(k-1,i-d) * num(i), pMin(k-1,i-d) * num(i)) \end{aligned} \tag{5}$$

$$\begin{aligned} pMin(k,i) = min(&pMax(k-1,i-1) * num(i), pMin(k-1,i-1) * num(i), \ldots, \\ &pMax(k-1,i-d) * num(i), pMin(k-1,i-d) * num(i)) \end{aligned} \tag{6}$$

综合公式 1和公式 5和公式 6, 得到状态转移方程：

$$\begin{cases} pMax(k,i) = pMin(k,i)num(i); k = 1, i = 1,2,\ldots,n \\ \begin{cases} pMax(k,i) = max(pMax(k-1,i-j)*num(i), pMin(k-1,i-j)*num(i)); \\ pMin(k,i) = min(pMax(k-1,i-j)*num(i), pMin(k-1,i-j)*num(i)) \end{cases} & k \neq 1, j = 1,2,\ldots,d \end{cases}$$

3 C++ 代码

```
#include<iostream>
using namespace std;
#define MAX 100
inline long long max(long long a,long long b){return a>b?a:b;}
inline long long min(long long a,long long b){return a>b?b:a;}
int main()
{
  int N,K,D;
  int num[MAX]={0};
  long long pMax[MAX][MAX]={0};
  long long pMin[MAX][MAX]={0};
  cin >> N;
  for(int i=1;i<=N;i++)
  cin >> num[i];
  cin>>K>>D;
  for(int k=1;k<=K;k++)
  {
    for(int i=1;i<=N;i++)
    {
      if(k==1)
        pMax[k][i]=pMin[k][i]=num[i];
      else
      {
        for(int j=i-1;i-j<=D&&j>0;--j)
        {
        pMax[k][i]=max(pMax[k][i],max(pMax[k-1][j]*num[i],pMin[k-1][j]*num[i]));
        pMin[k][i]=min(pMin[k][i],min(pMax[k-1][j]*num[i],pMin[k-1][j]*num[i]));
        }
```

```
        }
      }
    }
    long long result = 0;
    for(int i=1;i<=N;i++)
      result = max(result,pMax[K][i]);
    cout << result;
    return 0;
}
```