信奥算法

2728

请同学设计满分算法

有哪些可能的思路

边权平分为2个点权

原题需要求解两者差值
若边权不归任何人所有
等效于改边权两人平分

```
10      for(ll i=1;i<=n;++i){
11          scanf("%lld",&v[i]);
12          v[i]<<=1;
13      }
14      for(ll i=1;i<=m;++i){
15          ll a,b,c;
16          scanf("%lld %lld %lld",&a,&b,&c);
17          v[a]+=c;
18          v[b]+=c;
19      }
20      sort(v+1,v+1+n);
21      reverse(v+1,v+1+n);
22      ll A=0, B=0;
23      for(ll i=1;i<=n;++i)
24          if(i&1) A+=v[i];
25          else B+=v[i];
26      cout<<((A-B)>>1)<<endl;
```

涉及除法的问题
考虑转乘法

2719

手算

输入	输出？
5 2 1 5 7 6 9	5

修改后的序列
-1 3 5 6 9

最优解形式的简化

决策形式过于丰富

- 可以选择任意子段
- 可以选择增加或减少数值
- 可以选择改变量为[-x,x]内任意数值

简化形式

- 只考虑选某个前缀序列
  整体减去x
- 简化版决策一定包含最优解！

枚举决策点

选择哪个前缀 整体减x

枚举后修改数值
再求整个序列的LIS

暴力

```
19 void solveBF(){
20     for(ll i=1;i<=n;++i)b[i]=a[i];
21     ll ans=0;
22     for(ll i=1;i<=n;++i){
23         b[i]-=x;
24         ans=max(ans,LIS());
25     }
26     cout<<ans<<endl;
27 }
```

请写出时间复杂度
$O(n^2 \log n)$

多次求解LIS

```
11 ll d[N];
12 ll LIS(){    //b[]最长上升子序列=b[]不升子序列最小划分数
13     for(ll i=1;i<=n;++i)d[i]=INF;
14     for(ll i=1;i<=n;++i){
15         *lower_bound(d+1,d+1+n,b[i])=b[i];
16     }
17     return lower_bound(d+1,d+1+n,INF)-d-1;
18 }
```

LIS时间复杂度	整个程序时间复杂度
O(nlogn)	$O(n^2logn)$

如何加速？

原始序列 1 5 7 | 6 9 8

修改后的序列 -1 3 5 | 6 9 8

左侧前缀
预计算LIS

右侧后缀
预计算LIS

预计算

f[i]表示原始序列中以i号为右端点的LIS

g[i]表示原始序列中只改变i号数值为a[i]-x后
以i号为左端点的LIS

n=6, x=2

	i=1	i=2	i=3	i=4	i=5	i=6
a[i]=	1	5	7	6	9	8
b[i]=a[i]-x=	-1	3	5	4	7	6

预计算

f[i]表示原始序列中以i号为右端点的LIS

g[i]表示原始序列中只改变i号数值为a[i]-x后
以i号为左端点的LIS

n=6, x=2

	i=1	i=2	i=3	i=4	i=5	i=6
a[i]=	1	5	7	6	9	8
b[i]=a[i]-x=	-1	3	5	4	7	6

f[i]=

g[i]=

答案为 {f[i]+g[i]-1}的最大值

```
33 void solve(){
34     for(ll i=1;i<=n;++i)b[i]=a[i]-x;
35     for(ll i=1;i<=n;++i)d[i]=INF;
36     for(ll i=1;i<=n;++i){
37         ll id=lower_bound(d+1,d+1+n,b[i])-d;
38         d[id]=b[i];
39         f[i]=id;
40     }
```

反向 a[] 最长下降子序列
= 反向 -a[] 最长上升子序列
= 反向 -a[] 不升子序列最小划分数

```
41      for(ll i=1;i<=n;++i)d[i]=INF;
42      for(ll i=n;i>=1;--i){
43          ll id=lower_bound(d+1,d+1+n,-b[i])-d;
44          g[i]=id;
45          id=lower_bound(d+1,d+1+n,-a[i])-d;
46          d[id]=-a[i];
47      }
```

大小数据分离
确保部分分

```
53 int main(){
54     freopen("future.in","r",stdin);
55     freopen("future.out","w",stdout);
56     input();
57     if(n<=2000)
58         solveBF();
59     else
60         solve();
61     return 0;
62 }
```

2729

手算

输入	输出？
7 1 3 2 4 5 3 9	3

【说明】将第一个数值为3的减少为2，将第二个为3的数值增加到5，总代价为|2-3|+|5-3| = 3，并且修改过后序列为一个不下降序列1,2,2,4,5,5,9。

如何形成思路

尝试	序列问题利用二维平面点 可视化启发思路
尝试	n=1,2,3 小数据启发思路

n=3

```
10 void solve3(){
11     if(x[1]<=x[2]&&x[2]<=x[3]){ printf("0\n"); return; }
12     if(x[1]>=x[2]&&x[2]>=x[3]){ printf("0\n"); return; }
13     ll ans1=abs(x[2]-x[1]);
14     ll ans2=abs(x[2]-x[3]);
15     printf("%lld\n",
16 }
```

识别小难点

不升序列
不降序列

两种情况
本质对称
集中火力
解决一种

识别大难点

可行解太多	每个数可以取任意值

通过观察 提出猜想

最优解的 简化形式	最终序列的取值只考虑 原序列出现过的数值

增加了对解题有用的额外条件

证明思路：反证+调整

尝试DP

只算不降序列情况

请设计状态含义

A

f[i][j]表示修改后序列i号数
取值为原序列第j小数值时
前i位的最小代价

B

g[i][j]表示修改后序列i号数
取值为原序列前j小数值之一时
前i位的最小代价

```
/*
只算不降序列情况
另解:
f[i][j]表示修改后序列i 号数
取值为原序列第j 小数值时前i 位最小代价
n=7
      j=1, 2, 3, 4, 5, 6, 7
  x[j]=1, 3, 2, 4, 5, 3, 9
排序后
 xs[j]=1  2  3  3  4  5  9

 f表 j=1, 2, 3, 4, 5, 6, 7
  i=1  0  1  2  2  3  4  8
  i=2  2  1  0  0  1  2  6
  i=3
  i=4
  i=5
  i=6
  i=7

f[i][j]=min{f[i-1][1],f[i-1][2],..,f[i-1][j]}
        + abs(x[i]-xs[j])}
决策: 选哪个, 前缀最值
*/
```

```
/*
只算不降序列情况
g[i][j]表示修改后序列i号数
取值为原序列前j小数值之一时前i位最小代价
n=7
      j=1, 2, 3, 4, 5, 6, 7
  x[j]=1, 3, 2, 4, 5, 3, 9
排序后
 xs[j]=1  2  3  3  4  5  9

 g表 j=1, 2, 3, 4, 5, 6, 7
  i=1  0  0  0  0  0  0  0
  i=2  2  1  0  0  0  0  0
  i=3  3  1  1  1  1  1  1
  i=4  6  3  2  2  1  1  1
  i=5 10  6  4  4  2  1  1
  i=6 12  7  4  4  3  3  3
  i=7
*/
```

跑两次

```
73 void solve(){
74     for(ll j=1;j<=n;++j)xs[j]=x[j];
75     sort(xs+1,xs+1+n);
76     ll ans=minCost();
77     reverse(x+1,x+1+n);
78
79     printf("%lld\n",ans);
80 }
```

大小数据分离
确保部分分

```
92  int main(){
93      freopen("monotone.in","r",stdin);
94      freopen("monotone.out","w",stdout);
95      input();
96      if(n<=2)
97          printf("0\n");
98      else if(n==3)
99          solve3();
100     else
101         solve();
102     return 0;
103 }
```

太戈编程

2728,2719,2729