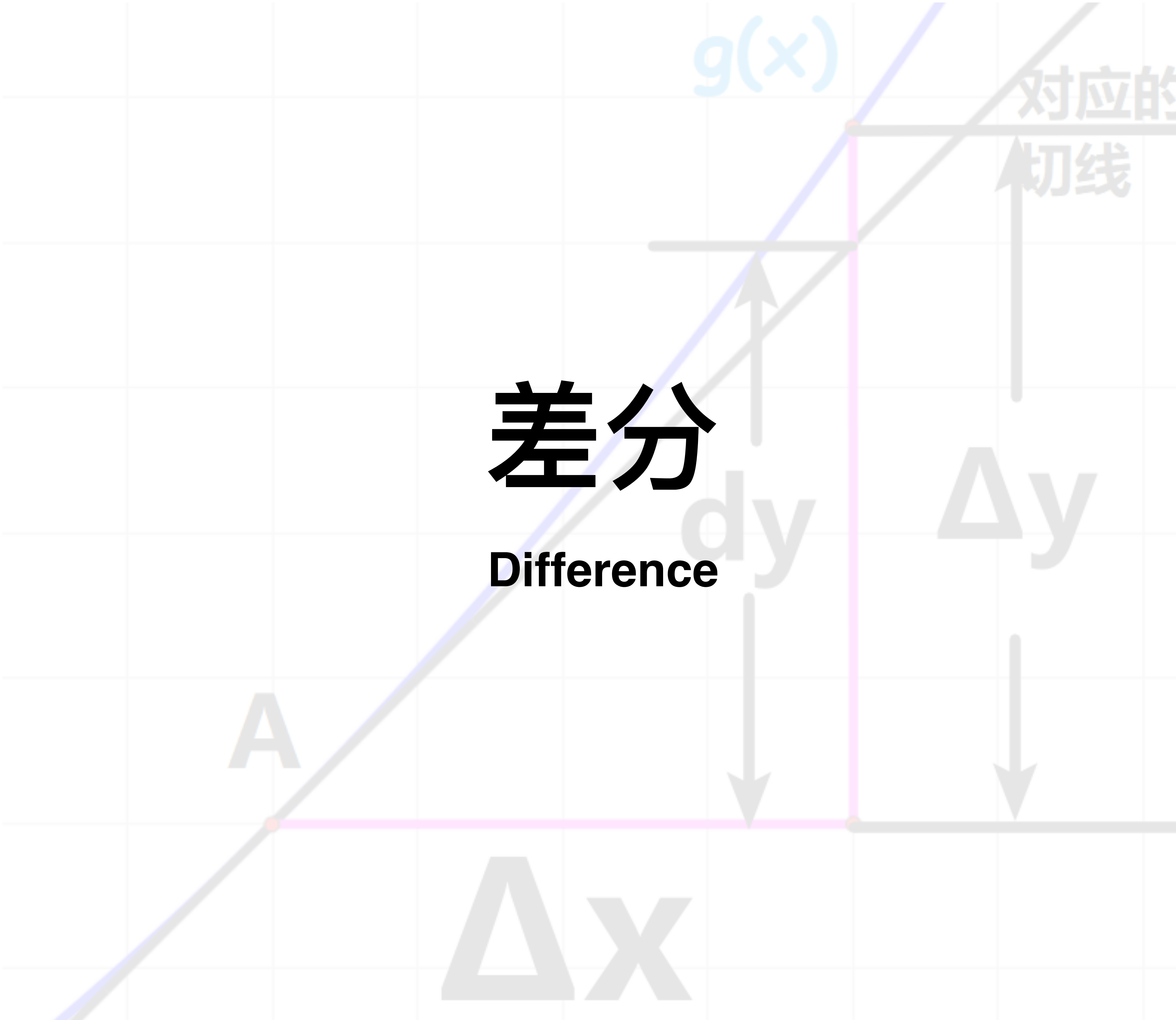
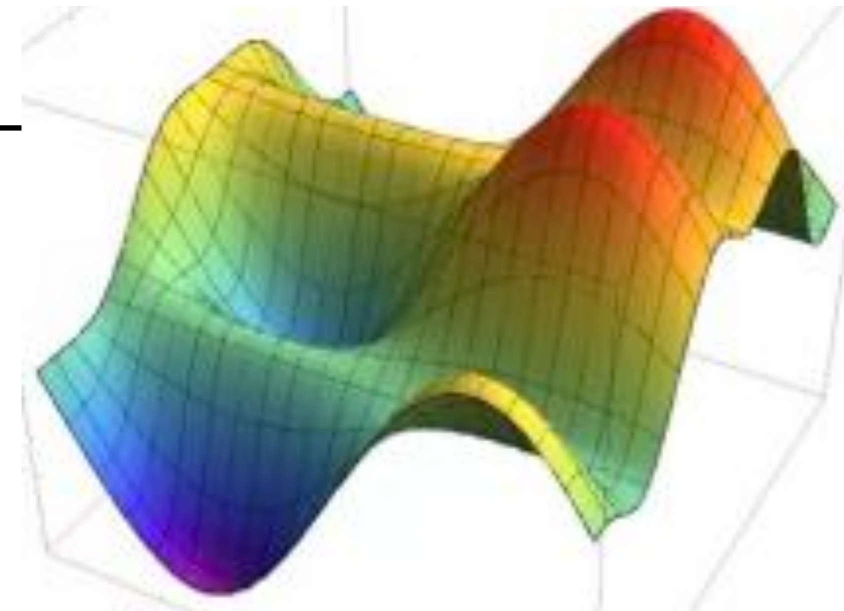
差分

Difference



装逼定义



→ **差分是一种局部差**

形式化表达为： △f=f(x+△x)-f(x)

也可理解为x→x+△x时F的**增量**

差分的逆运算是“前缀和”（增量和）

→ **△x→0时，差分→微分（Differenciation）**

df=f(x+dx)-f(x)

微分的逆运算是积分

$$\boxed{\int_0^A f(x)dx \equiv \lim_{\Delta x \to 0+} \sum_{i=0}^{A/\Delta x} f(i \cdot \Delta x) \cdot \Delta x}$$

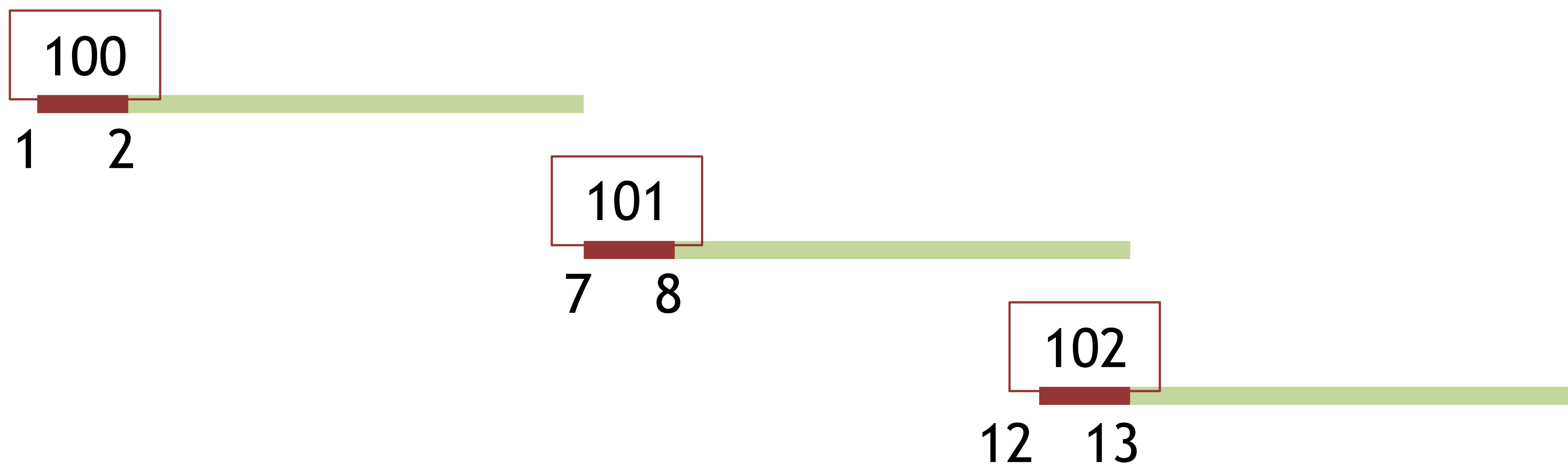
椅子危机

离线一维差分

椅子危机

n个公开课，第i个课从时刻si到时刻ti，要xi把椅子。椅子搬到另一课堂需要5分钟。请问至少需要几把椅子?

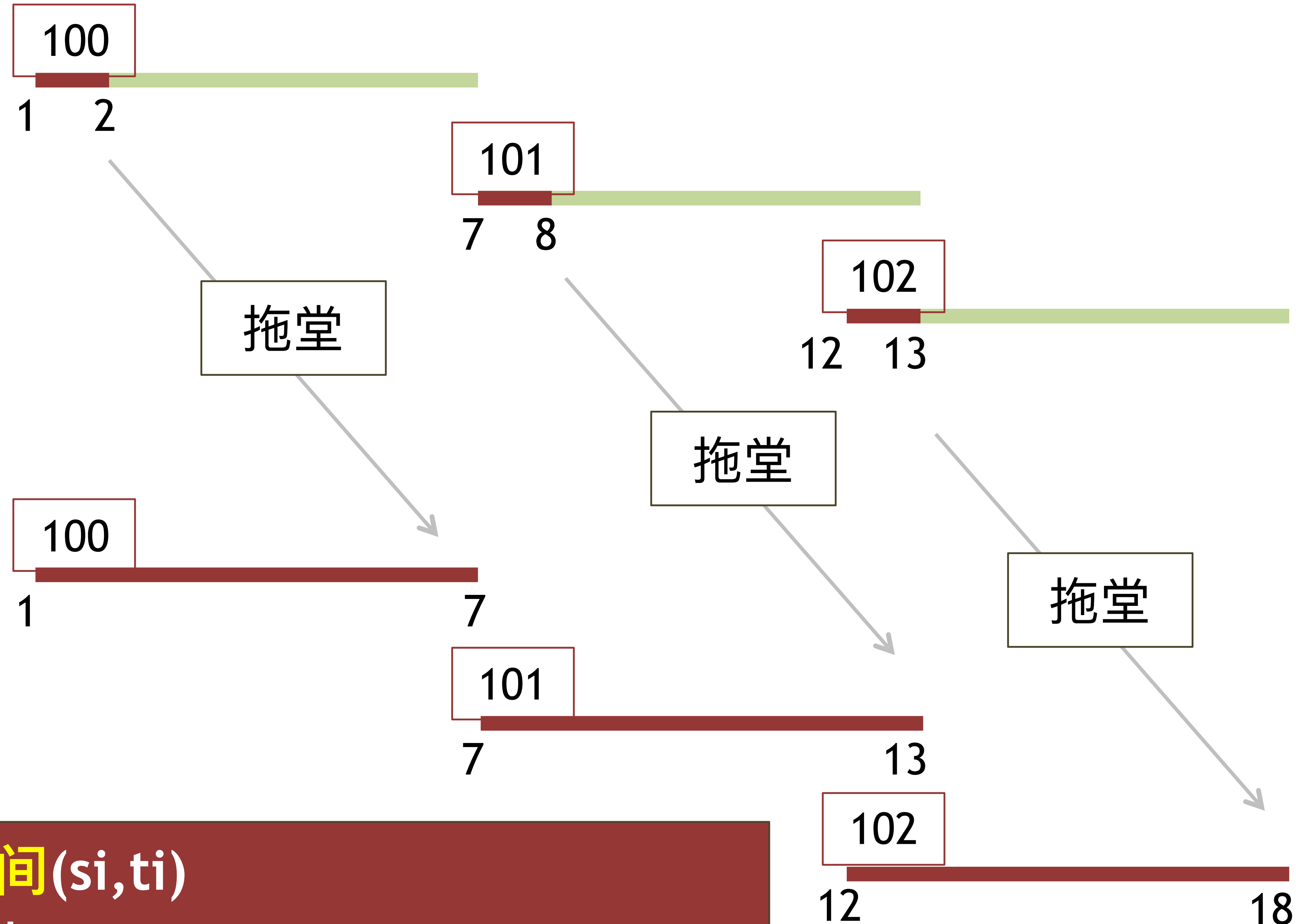
输入样例
3
1 2 100
7 8 101
12 13 102



输出样例
203

初步分析

输入样例
3
1 2 100
7 8 101
12 13 102



n个课程 → n个**区间**(si,ti)
需要椅子xi → 权值xi
搬动要5分钟 → ti+=5
最多同时需要多少椅子 → 区间带权**厚度**最大值

一维差分

f[i]=[i,i+1]的厚度
区间i影响O(n)处厚度

g[i]=f[i]-f[i-1]（厚度差分）
区间i影响2处厚度差

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1 2 100	+100						-100											
7 8 101							+101						-101					
12 13 102												+102						-102
g(i)	100	0	0	0	0	0	1	0	0	0	0	102	-101	0	0	0	0	-102
f(i)	100	100	100	100	100	100	101	101	101	101	101	203	102	102	102	102	102	0

先维护g数组，最后求一遍前缀和得到f，再求max即可

```cpp
const int R=1005;
int n,ans,g[1005];
int main(){
    cin>>n;
    for(int i=0,s,t,x;i<n;i++){
        cin>>s>>t>>x;
        g[s]+=x,g[t+5]-=x;
    }
    for(int i=0,s=0;i<R;i++) ans=max(ans,s+=g[i]);
    cout<<ans<<endl;
    return 0;
}
```

维护差分

求max(前缀和)

复杂度：O(n+R)

P667 最强大脑2

在线一维差分

破题

→ **在线区间问询**（段更新，点查询）

→ **暴力：** O(nm)

→ **线段树：** O(mlogn)（需要Tag-Lazy）

→ **BIT：** 不支持段更新

差分一下（裂项）： **b[i]=a[i]-a[i-1]**（取a[0]=0）

逆运算： a[i]=(a[i]-a[i-1])+(a[i-1]-a[i-2])+…+(a[1]-a[0])

=b[i]+b[i-1]+…+b[1]（前缀和）

段更新 → 2*点更新

点查询 → 前缀和查询

段更新 → **1维差分** → **点更新**

→ **在线前缀和查询，BIT正合适**

差分

a： 10 20 30 2 3 15 20 7

b： 10 10 10 -28 1 12 5 -13

[1~4]+=5

a： 15 25 35 7 3 15 20 7

b： 15 10 10 -28 -4 12 5 -13

[3~7]+=10

a： 15 25 45 17 13 25 30 7

b： 15 10 20 -28 -4 12 5 -23

[4~8]+=10

a： 15 25 45 27 23 35 40 17

b： 15 10 20 -18 -4 12 5 -23

a的段更新=d的2次点更新

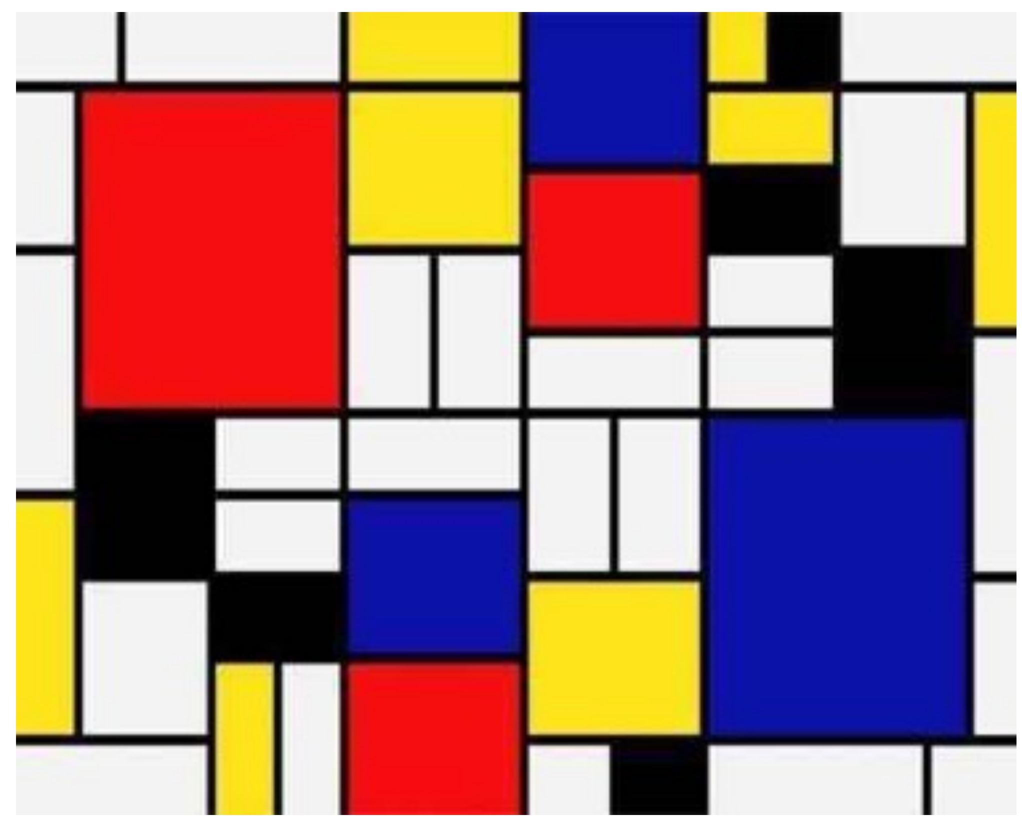
a的点查询=d的前缀和查询

$\boxed{\text{差分数组上建BIT}}$

P1486 蒙德里安

离线二维差分

蒙德里安（Piet Cornelies Mondrian，1872~1944），荷兰画家，风格派运动幕后艺术家和非具象绘画的创始者之一，对后世的建筑、设计等影响很大。蒙德里安是几何抽象画派的先驱，以几何图形为绘画的基本元素，与杜斯堡等创立了“风格派”，提倡自己的艺术“新造型主义”。他还认为艺术应根本脱离自然的外在形式，以表现抽象精神为目的，追求人与神统一的绝对境界，也就是现在我们熟知的“纯粹抽象”



现在你要临摹蒙德里安的画风作画。画布可以看做是一个n*n的棋盘。你一共画了p个矩形，第i个矩形的左上角是第ai行第bi列，右下角是第xi行第yi列。当然了，有些矩形可能重叠也是很正常的。请问还有几格没有被覆盖到?

【输入格式】
第一行2个正整数n，p。n≤1000，p≤200000
后面p行每行四个正整数代表ai,bi,xi,yi，均不超过n

【输出格式】
一个非负整数表示未被盖到的格子数

输入样例	输出样例
3 2 1 1 2 2 2 2 3 3	2

国王的奖赏5?

→ **沿x方向扫描**

y方向上覆盖状态**仅在矩形的竖边处有变化**

→ **y方向建线段树，矩形左右边界排序**

左边界段更新+1，右边界段更新-1

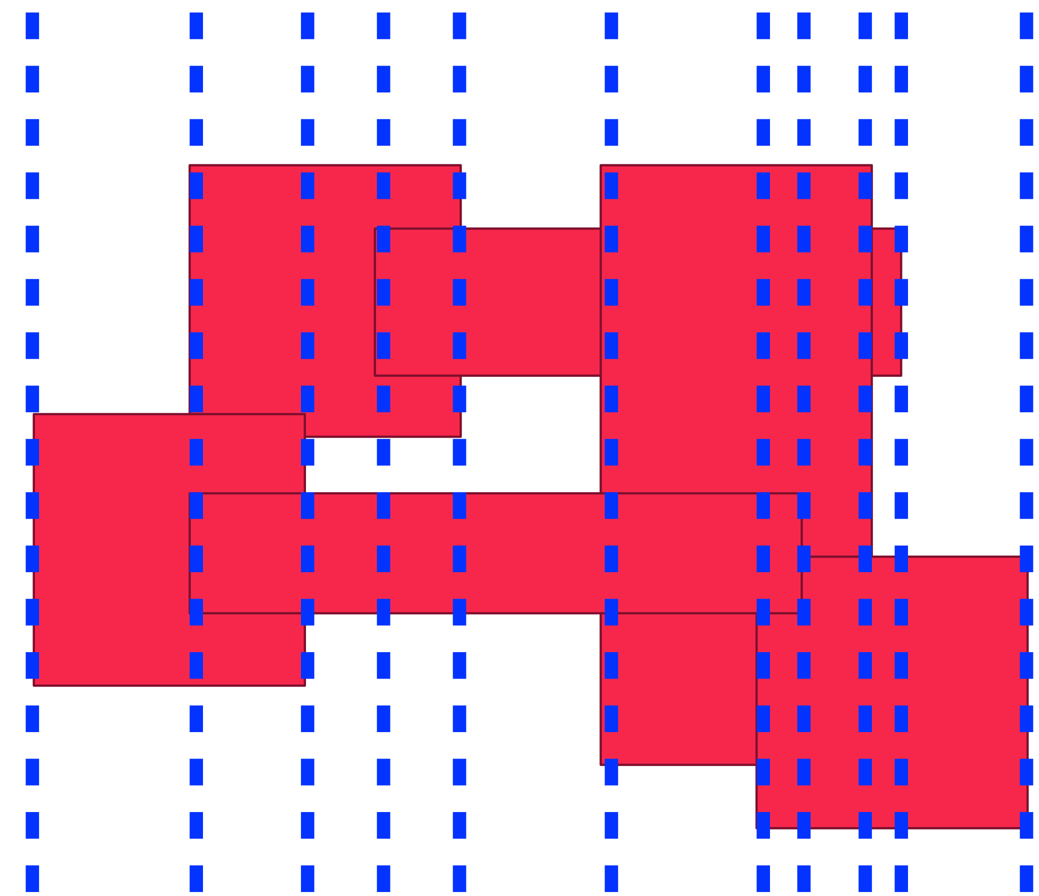
目标函数：y方向**并集长度**

复杂度：O(plogn+plogp))，**空间**：O(n)

→ **其实所谓“y方向上覆盖状态变化”**

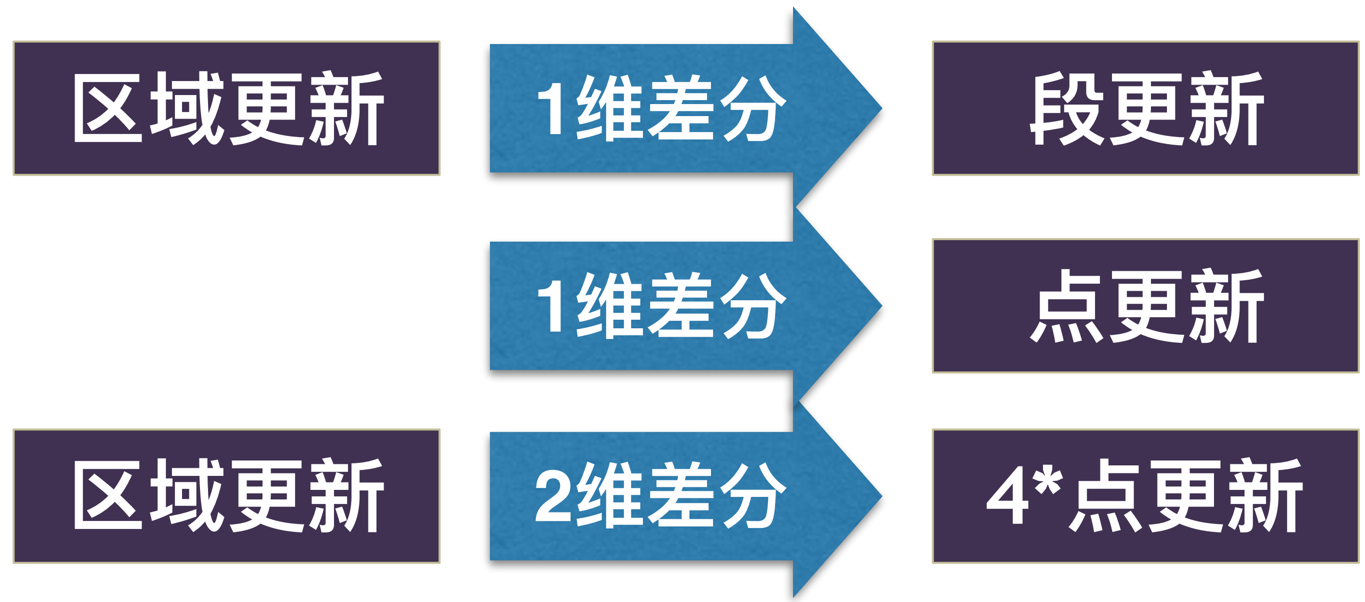
就是**x方向上的差分（偏差分）**

b[i,j]=a[i,j]-a[i,j-1]



二维差分

a[i-1][j-1]	a[i-1][j]
a[i][j-1]	a[i][j]



→ **能否直接变为点更新呢?**

b[i,j]=a[i,j]-a[i-1,j]-a[i,j-1]+a[i-1,j-1]

→ **区域更新：** a[(i1,j1)-(i2,j2)]+=x

b[i1,j1]+=x

b[i2+1,j2+1]+=x

b[i1,j2+1]-=x

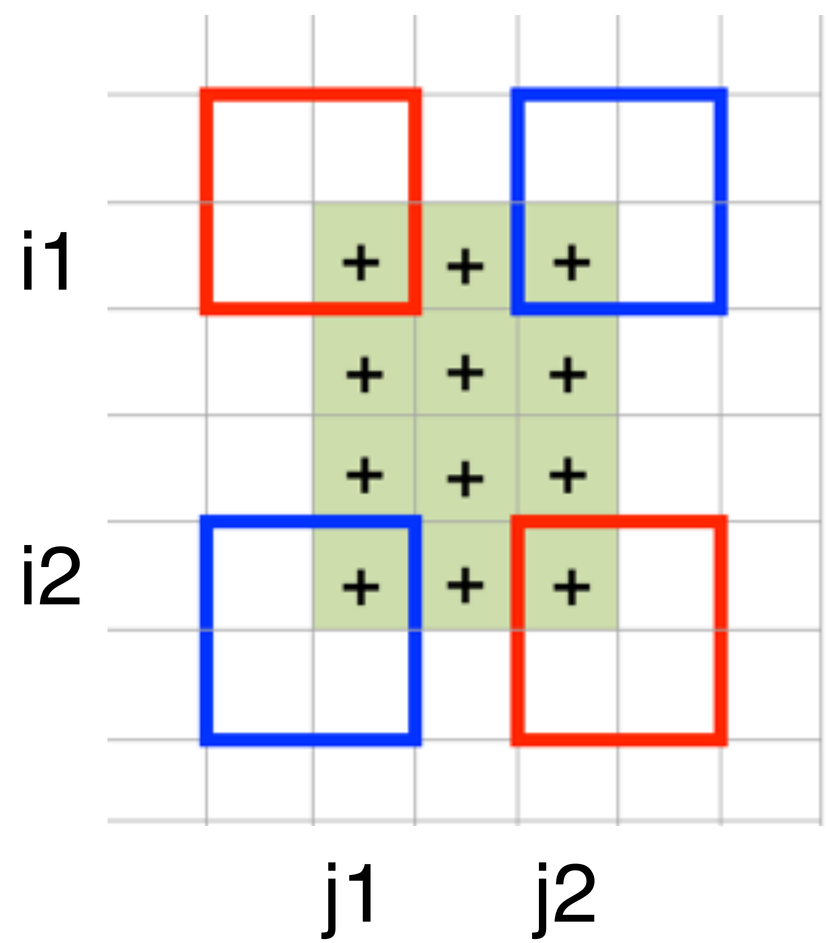
b[i2+1,j1]-=x

Over

→ **逆运算**

a[i,j]=sum {b[(1,1)-(i,j)]} （二维前缀和）

请自行证明



```
for(int i=1;i<=p;i++){
    cin>>a>>b>>x>>y;
    d[a][b]++;
    d[a][y+1]--;
    d[x+1][b]--;
    d[x+1][y+1]++;
}
```

二维差分维护区域更新

复杂度：O(p+n^2)

对比：O(plogn+plogp)

p>>n时，比线段树更快

```
int ans=n*n;
for(int i=1;i<=n;i++)
    for(int j=1;j<=n;j++){
        s[i][j]=d[i][j]+s[i-1][j]+s[i][j-1]-s[i-1][j-1];
        if(s[i][j]) ans--;
    }
```

二维前缀和恢复原数组

补集转化

拓展

→ **国王的奖赏5原题坐标范围很大（~1e9）**

所以原题还需要先做离散化

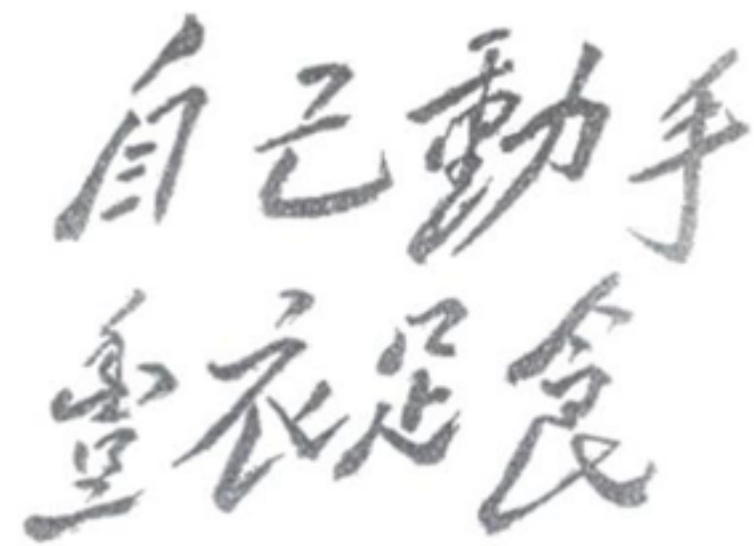
正解： 离散化+转动态+线段树

建议尝试： 离散化+二维差分

→ **在线二维差分?**

二维在线问询： 区域更新+区域sum查询

建议尝试： 差分+二维BIT实现↑



P866 战略轰炸7

离线树上差分（1.5维？）

建模

→ **破题（计数问题）**

图上有两类边：一类形成生成树🤔，另一类无限制

删除两类边**各一条**，使图不连通。求方案数

→ **暴力算法**

枚举删边，判断连通性（DFS/BFS/并查集）

复杂度：$O(nm(n+m))$（40分）

→ **你能写对吗**😏

这个暴力要写得好也是要点技巧的

优化

→ **减少枚举量**

枚举一条边，另一条边是**剩余图中的桥**（Bridge）

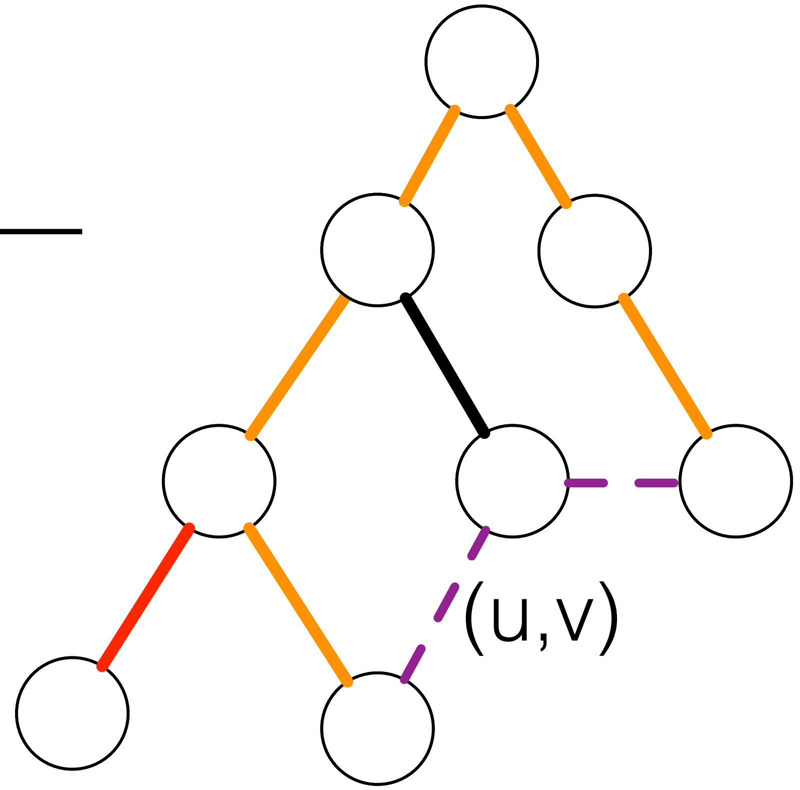
tarjan算法，求所有的桥，再判断下边的类型即可

→ **复杂度：**O(n(n+m))（60分）

→ **继续优化的余地**

还有用到一类边是**树**的特性

一般情况



→ **每条电缆(u,v)**

设u,v在树上的通路为R(u,v)

→ **如删R(u,v)上的光纤**

要不连通，**电缆只能删(u,v)**（必要条件）

否则用(u,v)替代被删的光纤，仍是一个完整的生成树

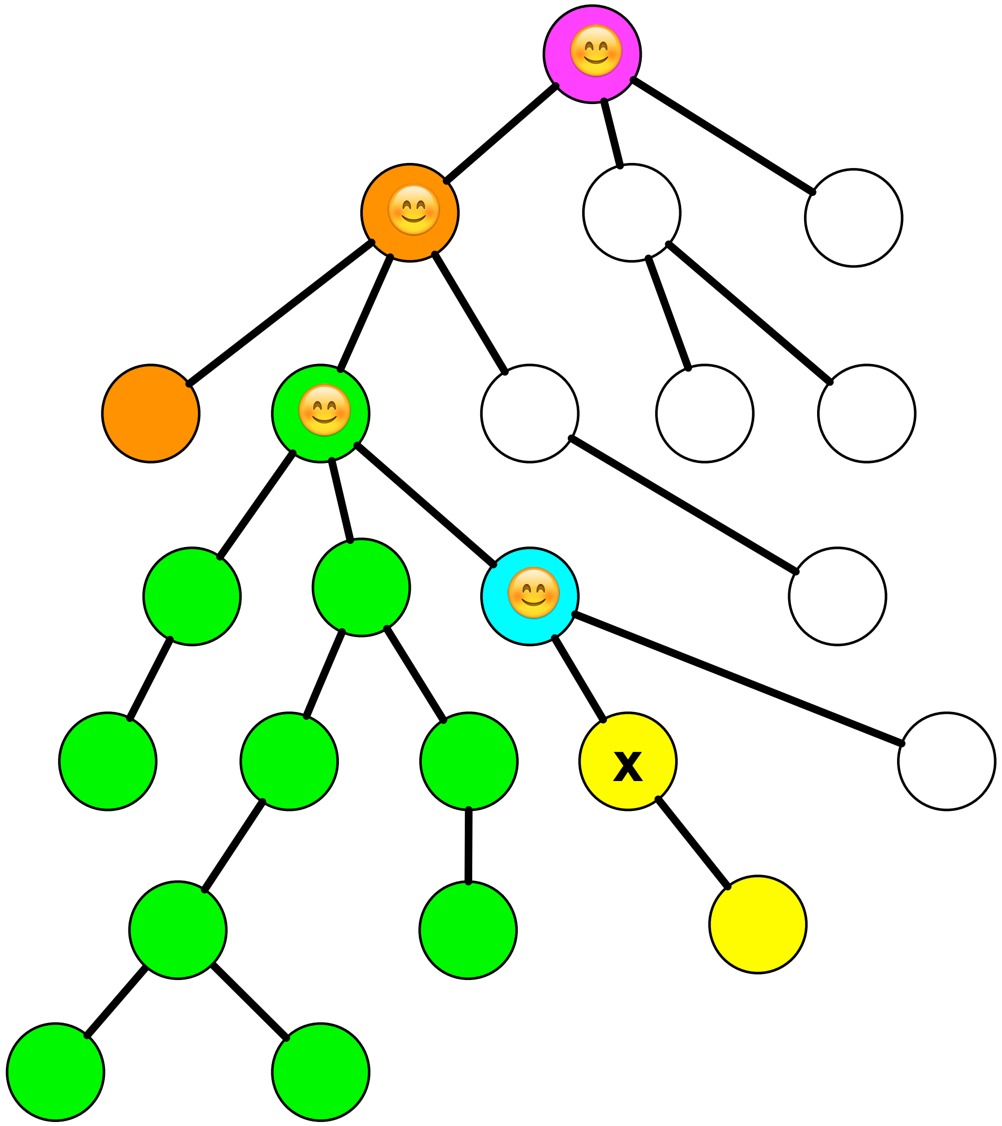
→ **统计每条光纤(x,y)在多少个R(u,v)上**🤔

1.ans=0，则(x,y)是桥，再删任意电缆都可

2.ans=1，只能再删唯一的那个环上的那条电缆

3.ans≥2，删(x,y)无解

离线LCA（选学）



→ **首先要求所有非树边端点的LCA**

你可以用倍增的方法，O((n+m)logn)

这里介绍**离线批量LCA**更快的方法**（tarjan算法）**

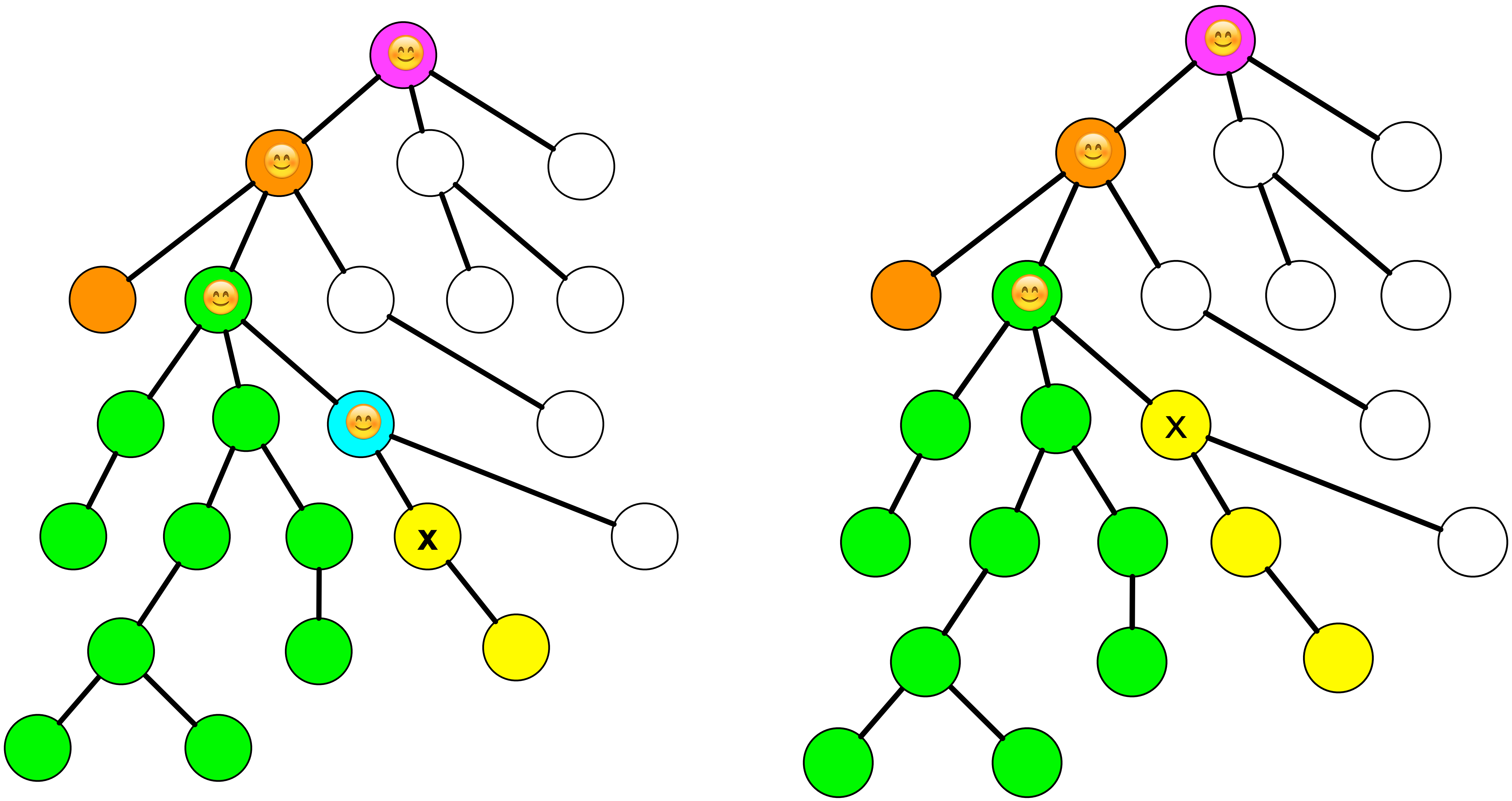
→ **按先序遍历顺序计算LCA**

当遍历到x时，**x与其他点的LCA一定为x的祖先**（废话）

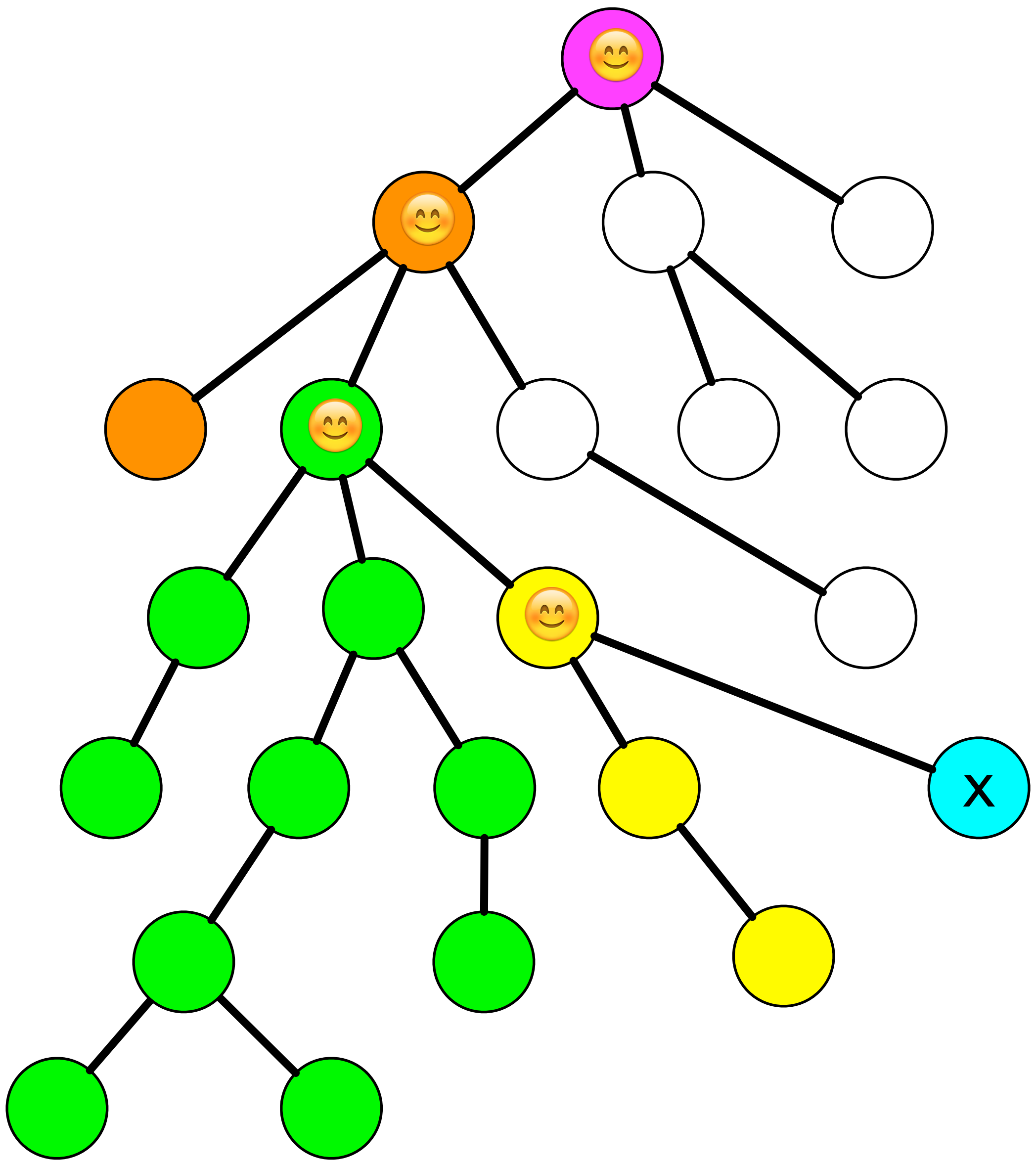
其他点根据与x的LCA不同分成**若干组**🤔

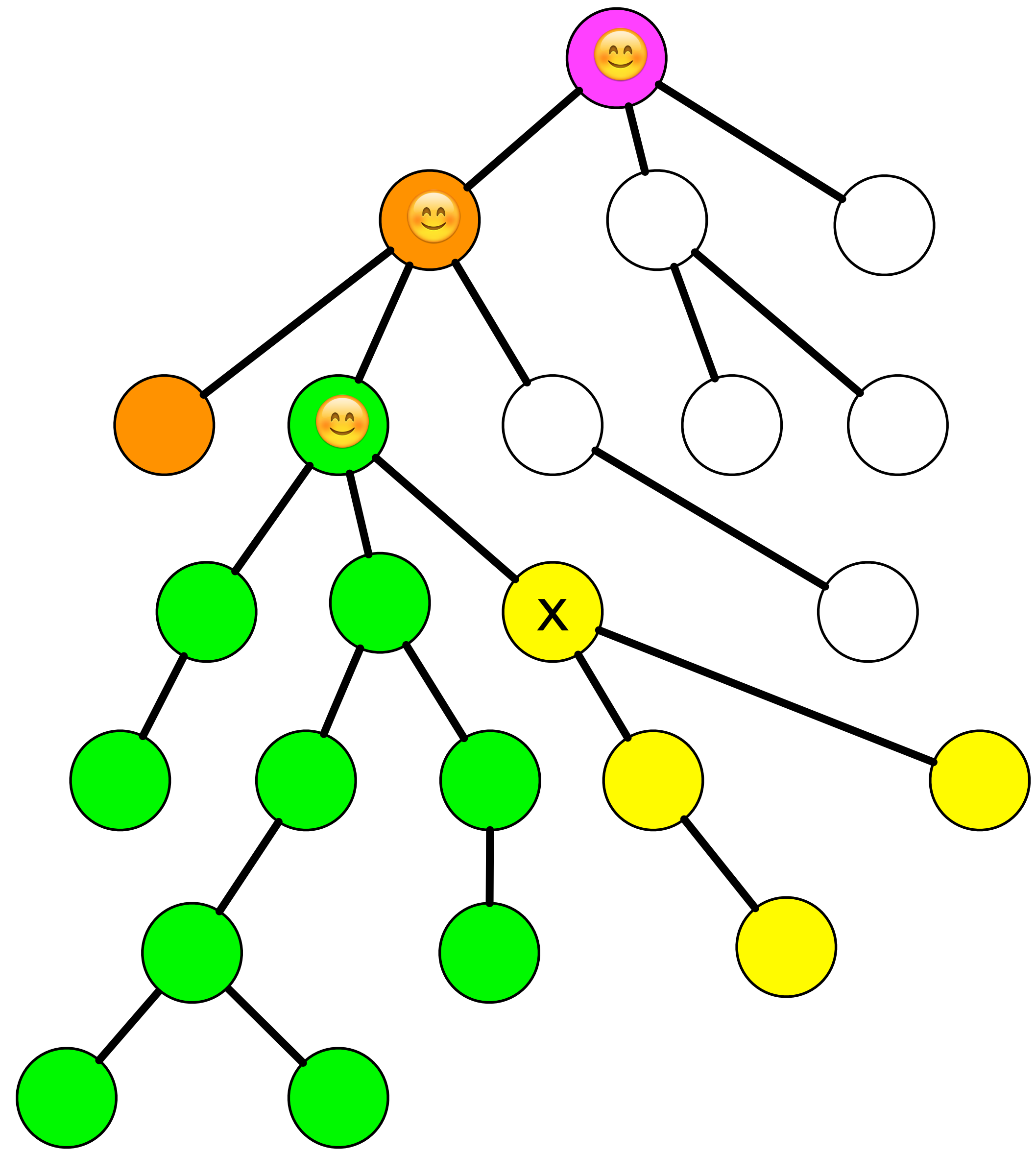
组长即为各LCA（x的祖先）🤔

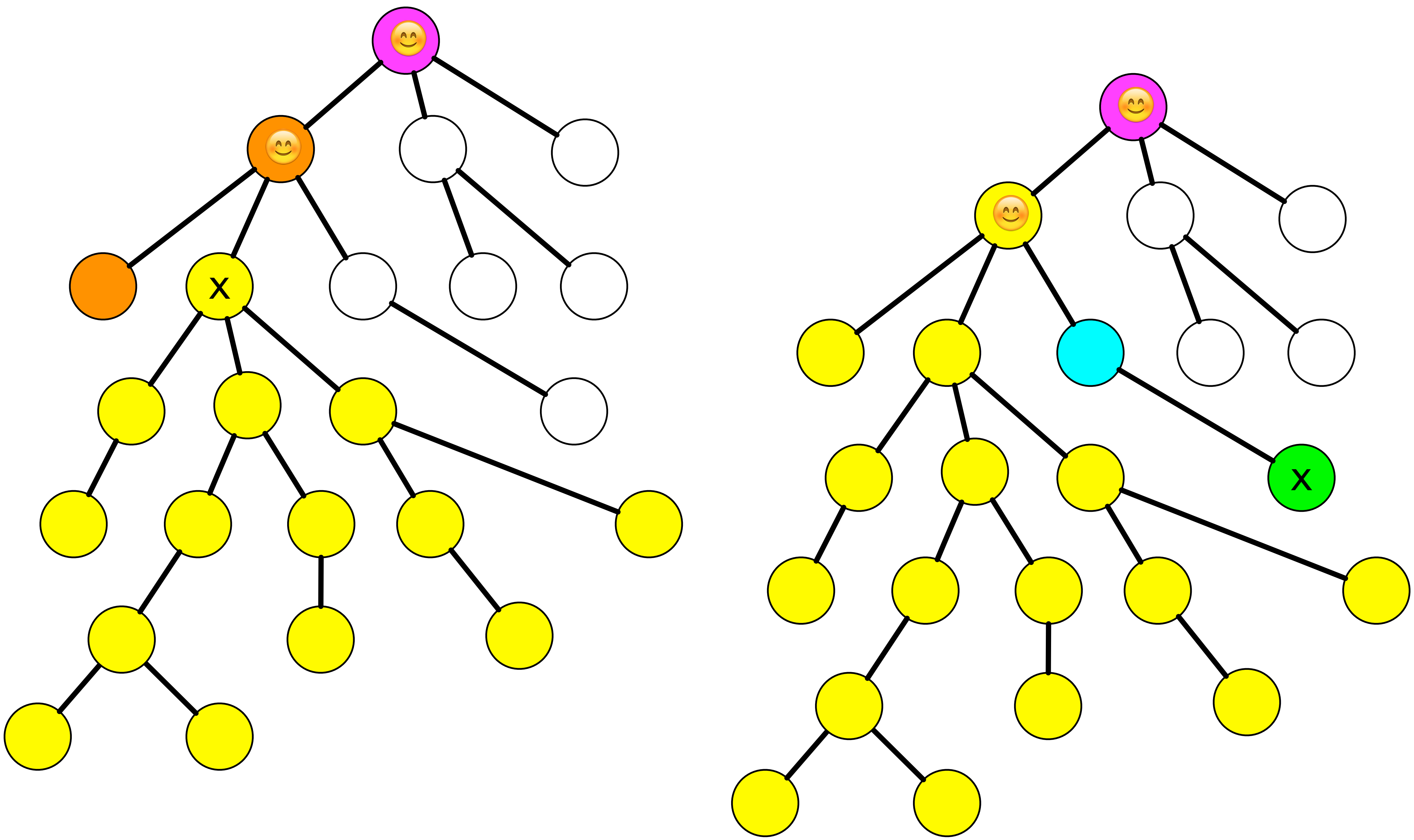
→ 当x回溯的时，分组只会合并🤔



→ 当x递归时，产生新分组🤔






x
x

```
int find(int x){
    return p[x]==x? x : p[x]=find(p[x]);
}

void tarjan(int x,int fa) {
    p[x]=x;
    for(int i=hd_g[x];i;i=es[i].nxt)
        if (es[i].t!=fa)
            tarjan(es[i].t,x),p[es[i].t]=x;
    vis[x]=1;
    for(int i=hd_q[x];i;i=es[i].nxt)
        if (vis[es[i].t])
            lca[(i-2*n+2+1)/2]=find(es[i].t);
}
```

贼犀利的代码👍

边表复用要
减去前图的部分

→ 分组变化过程显然是并查集略

复杂度：O(n+m)

每条光纤(x,y)在多少个环R(u,v)上

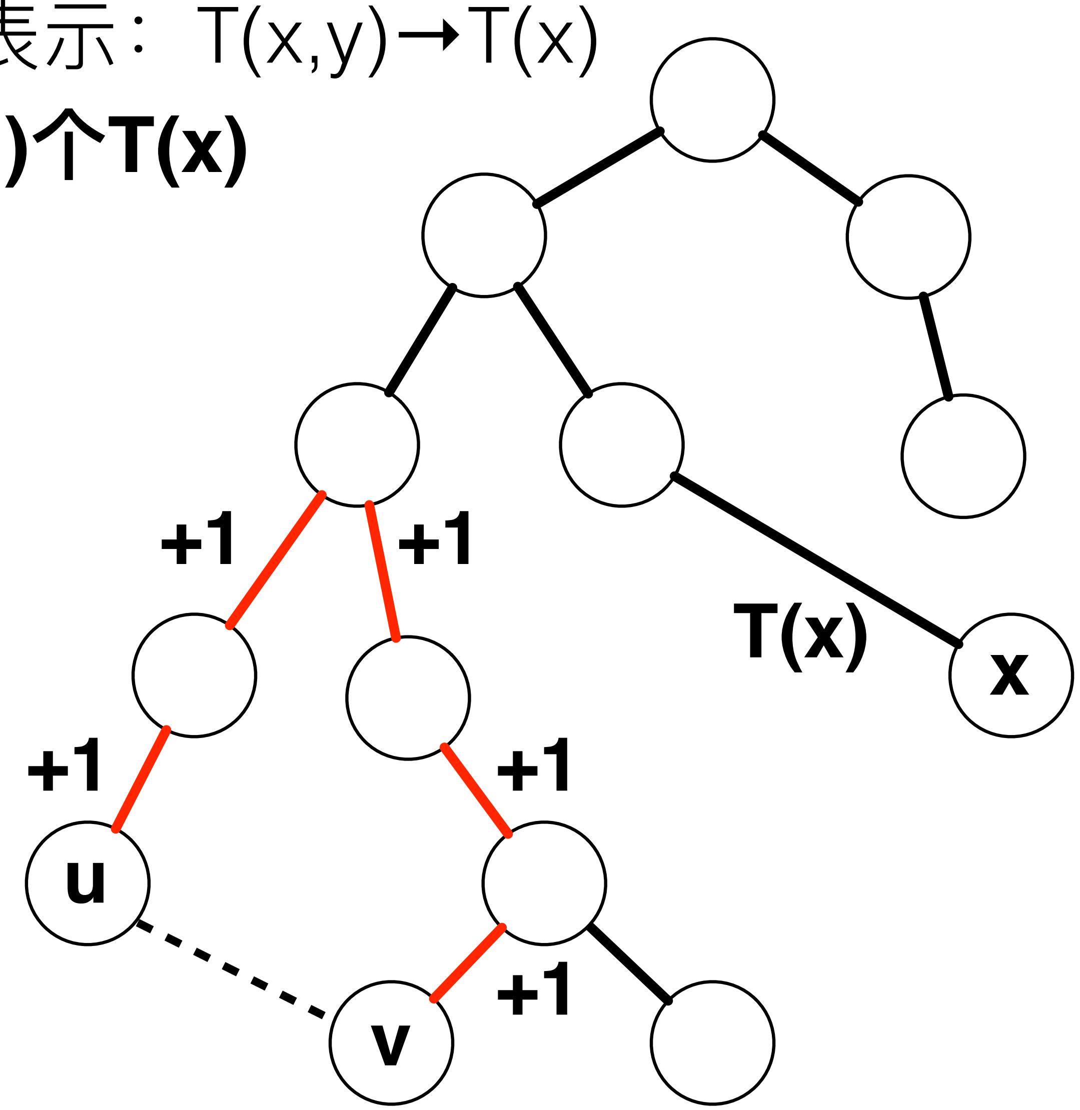
→ **等价于求所有R(u,v)在每条(x,y)重叠的厚度T(x,y)**

由于树上就n-1条边

树上的边可用深度较深的节点表示：T(x,y)→T(x)

→ **一个环R(u,v)影响d(u,v)=O(n)个T(x)**

d(u,v)是u,v在树上的距离



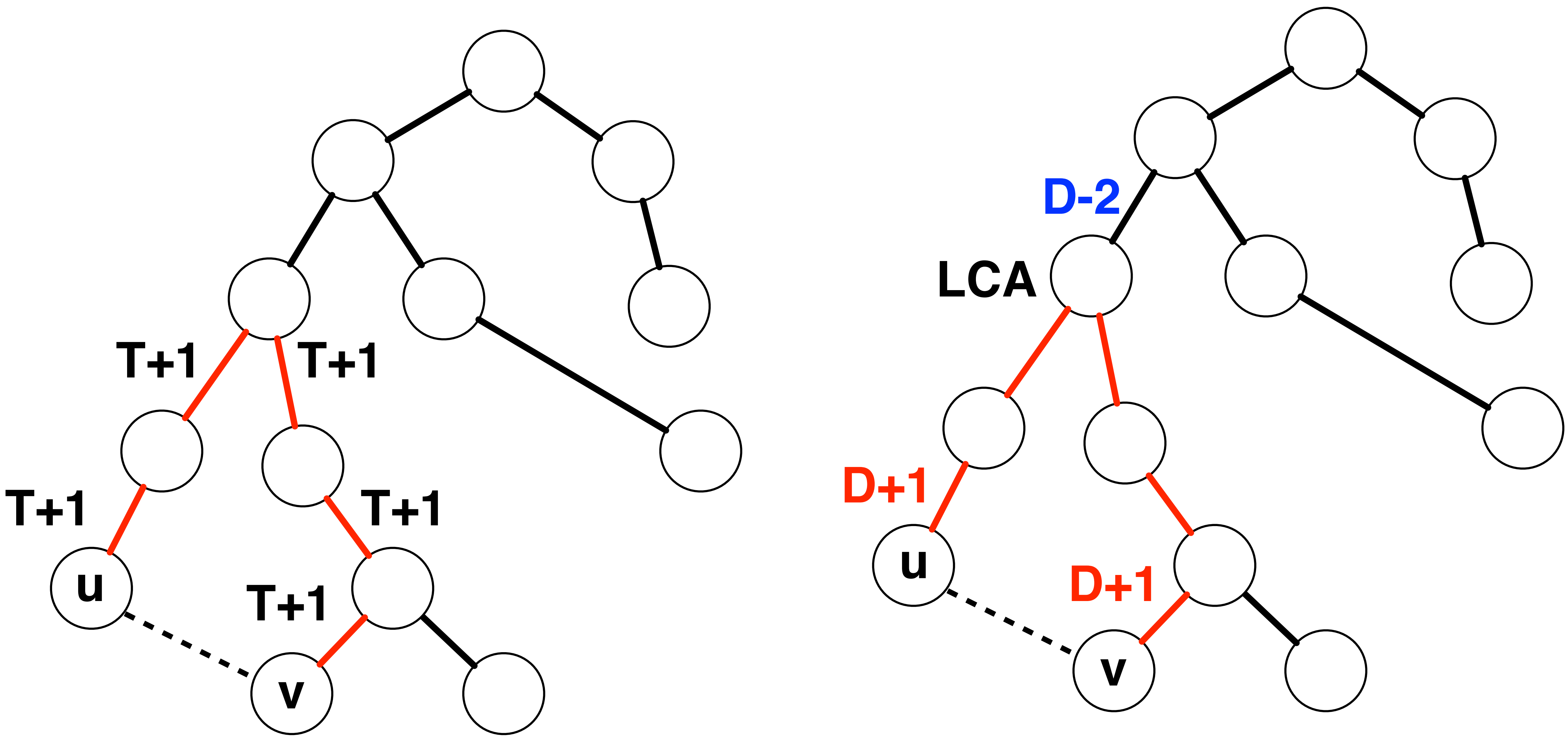
路径更新 → 树上差分 → 3*点更新

树上差分

→ D(x)=T(x)-sum{T(y)}，y是x的子节点**（爸爸减儿子）**

一次路径更新，D上受影响3个节点

→ **逆运算**：T(x)=sum{D(y)}，y在x的子树内**（子树和）**



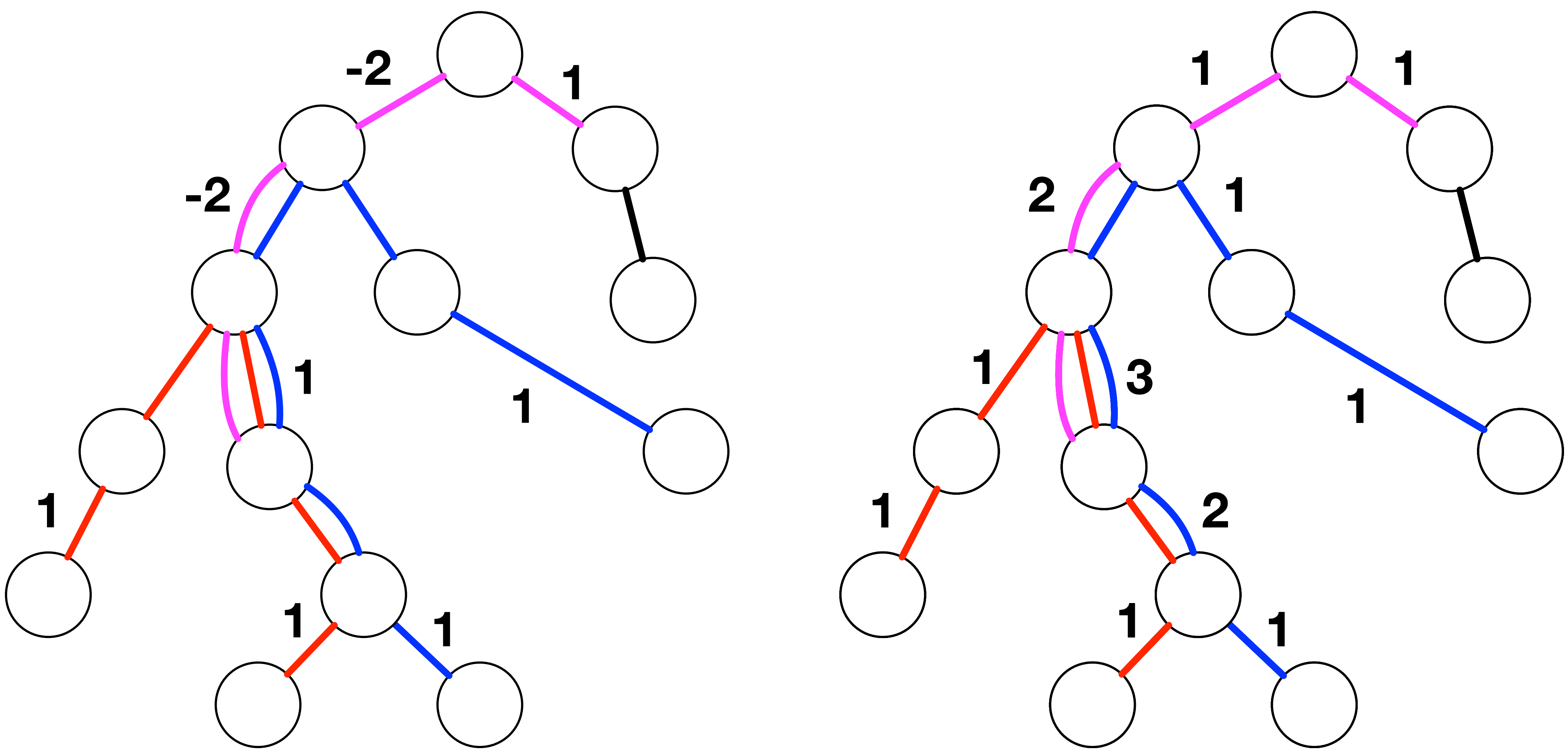
树上路径厚度

1. 每条路径(u,v)： D(u)++,D(v)++,D(LCA(u,v))-=2

2. 自底向上求一遍子树和，得到T(x)

复杂度： O(nlogn+mlogn+n)

(建倍增索引+更新D+DFS遍历求T)



```
  for(int i=1,a,b;i<=m;i++){
      a=es[n*2-2+i*2].t,b=es[n*2-2+i*2-1].t;
      f[a]++,f[b]++,f[lca[i]]-=2; // 差分
  }
  dfs(1,0);

void dfs(int x,int fa){ // 差分求和
    for(int i=hd_g[x];i;i=es[i].nxt)
        if (es[i].t!=fa)
            dfs(es[i].t,x),f[x]+=f[es[i].t];
    // 覆盖0次则次要边随意，覆盖1次则只能切对应次要边
    ans+=(x!=1)*(m*(f[x]==0)+(f[x]==1));
}
```

复杂度： O(nlogn+mlogn+n)
（建倍增索引+更新D+DFS遍历求T）

拓展

→ **本题还可优化到线性**

注意logn是因为要计算LCA

而离线LCA有线性算法（Tarjan算法）

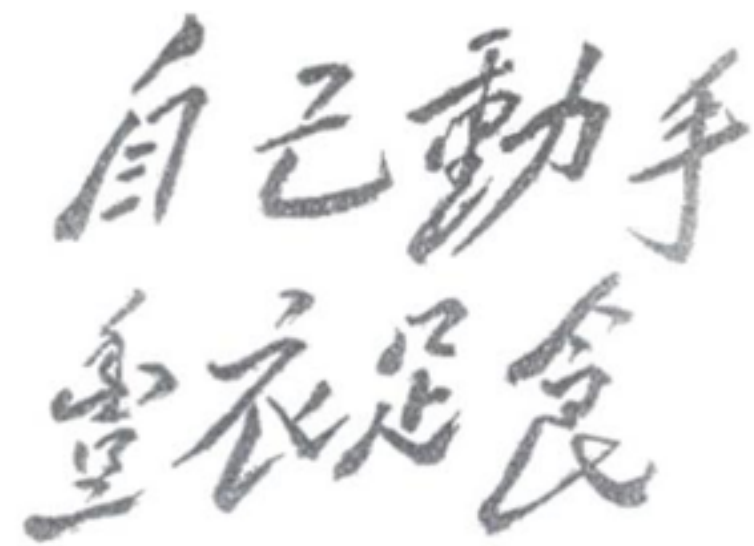
建议尝试： 自己去查LCA的tarjan算法

→ **在线树上差分**

树上在线问询： 路径更新+点查询

建议尝试： 树上差分+dfn序+BIT实现↑

在线LCA：倍增、欧拉序+RMQ(ST表)



P1439 运输计划

NOIP2015提高组T6

破题

请同学简述题意
突出核心要点

最优化
问题

无根树，有边权
决策对象： 将一条边权清为0（魔法）
约束条件： 无
优化目标： 给定的m条路径，总长最大值最小

破题

测试点编号	n=	m=	约定
1	100	1	
2		100	第 i 条航道连接 i 号星球与 i+1 号星球
3			
4	2000	1	
5	1000	1000	第 i 条航道连接 i 号星球与 i+1 号星球
6	2000	2000	
7	3000	3000	
8	1000	1000	
9	2000	2000	
10	3000	3000	
11	80000	1	
12	100000		
13	70000	70000	第 i 条航道连接 i 号星球与 i+1 号星球
14	80000	80000	
15	90000	90000	
16	100000	100000	
17	80000	80000	
18	90000	90000	
19	100000	100000	
20	300000	300000	
所有数据			$1 \le a_i, b_i, u_j, v_j \le n, 0 \le t_i \le 1000$

n^3

链

n^2

一次路径max问询(简单)

部分分（本页提高组的你应该很溜才行）

→ **m=1（BFS+求最长边）**：O(n)（20分）

→ **暴力（枚举零边+BFS）**：O(mn^2)（20分）

→ **优化**

BFS预计算所有路径(ui,vi)原始长度

枚举零边(x,p(x))，判断该边在哪些路径(ui,vi)上

判据：ui或vi在子树x中 && d(x)>d(LCA(u,v))

在子树中：预计算dfn，子树是连续一段（P1011祖孙问询）

复杂度：O(nm+nmlogn)（60分）

涉及LCA、dfn等树上基本操作，建议每个人都写一下

二分答案

→ **全局最优化（最大值最小），你懂的**

OK(L)=m条路径总长都不超过L，是否可行

→ **零边的条件**

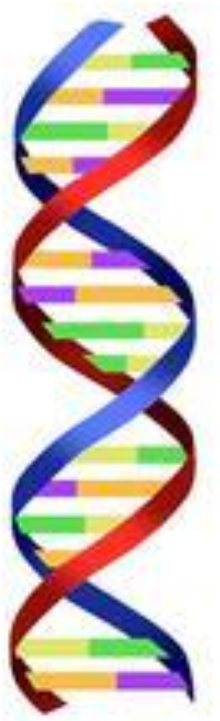
1. **零边原长≥dt≡原最长路径-L**（必要条件）

2. **零边在所有原长>L的路径上**（必要条件）

→ **不难证明1,2加一起就是充分的**😏

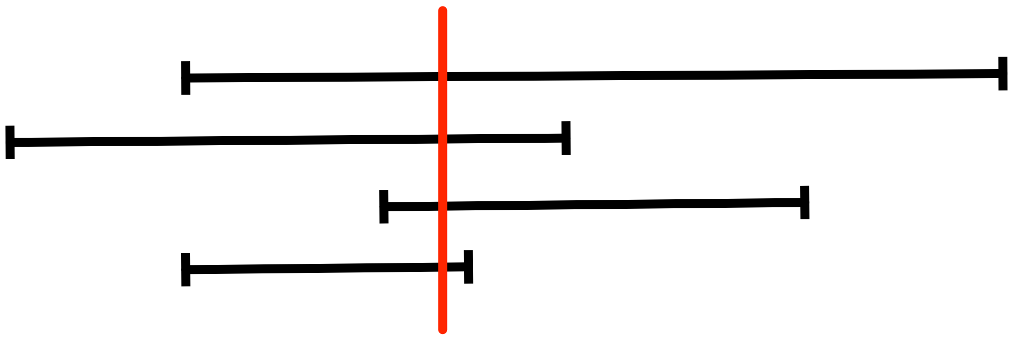
OK(L)=true的充要条件：

原长>L的所有路径经过同一条边，且此边长度≥dt



原长>L的所有路径经过同一条边，且此边长度≥dt

可行性问题



→ **设原长度>L的路径有tot条**

路径经过同一条边 <=> **厚度=tot**

→ **链状**

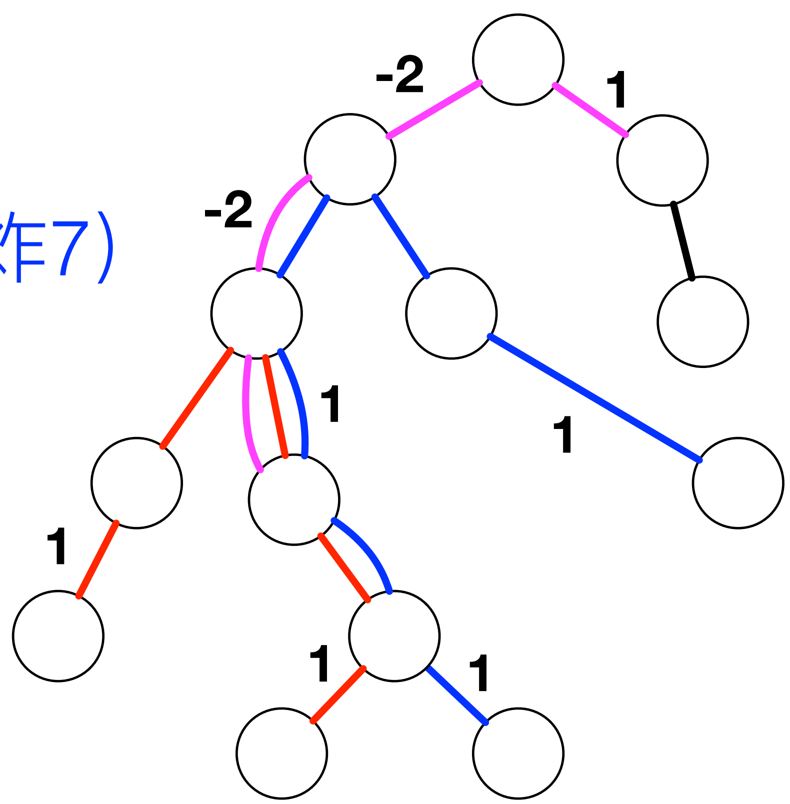
OK函数：求一维区间厚度（椅子危机）

一维差分：O(log(nT)*(m+n))（40分）

→ **树状**

OK函数：求树上路径一维区间厚度（战略轰炸7）

树上差分：O(log(nT)*(m+n))（100分）



一种感觉（多少人有同感的？）

→ **差分（及其逆运算）是一种“边界”与“区域”的转化**

有时候边界上好做，有时候区域上好做

对整个区域的同等操作，在差分上往往可以降维

对单点的原始值查询，在差分上往往会升维

差分

差分

作业

1.蒙德里安（P1486）
2.战略轰炸7（P866）
3.运输计划（P1439，至少60分）
4.国王的奖赏5（P862，选做，用离散化+二维差分）