差分数组

数列游戏 NKOJ3754

给定一个长度为N的序列：

首先进行X次操作，每次操作在Li和Ri这个区间加上一个数Ci。

然后进行Y次询问，每次询问Li到Ri的区间和。

初始序列都为0。

1<=N<=1000000,1<=X<=N, X<=Y<=N

1<=Li<=N,Li<=Ri<=N,|Ci|<=100000000000000

线段树裸题！

有没有其它简便的解法？

差分数组！

差分数组(差分数列)

对于一个数组A[ ]，其差分数组D[i]=A[i]-A[i-1] (i>0)且D[0]=A[0]

令**SumD[i]=D[0]+D[1]+D[2]+...+D[i]** （SumD[ ]是差分数组D[ ]的前缀和）

则**SumD[i]=A[0]+A[1]-A[0]+A[2]-A[1]+A[3]-A[2]+...+A[i]-A[i-1]=A[i]**

即A[i]的差分数组是D[i]， 而D[i]的前缀和是A[i]

对于“数列游戏”这题:

如果每次修改都修改从L到R的值的话，一定会TLE。

注意特殊处：这道题是先进行整体区间修改，最后才统一查询。

所以，我们只要维护一个差分数组就行了。

维护差分数组，对于将区间[L,R]加C，我们只需要将**D[L]+C**和**D[R+1]-C**

当修改完毕后，我们先求一遍差分前缀和就得到了修改后的数组A[ ]，

然后再对A[ ]求一遍前缀和，这样每次查询的时候只要计算一次就可以得到结果了

//参考代码

```
cin>>N>>X>>Y;
for(i=1;i<=X;i++)
{
        cin>>L>>R>>C;
        D[L]=D[L]+C;
        D[R+1]=(D[R+1]-C);
 }
 for(i=1;i<=N;i++)A[i]=(D[i]+A[i-1]);                    //D[i]=A[i]-A[i-1];
 for(i=1;i<=N;i++)SumA[i]=SumA[i-1]+A[i];
 for(i=1;i<=N;i++)
 {
        cin>>L>>R;
        cout<<SumA[R]-SumA[L-1]<<endl;
 }
```

数列操作 NKOJ3786

给定一个长度为n的数列{ A1,A2...An }，每次可以选择一个区间[L,R]，使这个区间内的数都+1或者都-1。

问至少需要多少次操作才能使数列中的所有数都一样,并求出在保证最少次数的前提下,最终得到的数列有多少种。

n=100,000，0<=Ai<2147483648

数列操作 解题分析

由于操作都是区间操作，那么区间里的数相对差值是不变的，那么我们可以考虑构建 **差分数组:D[i]=A[i]-A[i-1]**

显然，当**D数组除了第1个元素外所有其他元素都为0**时，A数组才会满足题目要求。
即最后所有A数组中所有元素的值都是A[1]

讨论一次操作对D数组的影响：

假设对A数组的区间**[L,R]整体+1**，那么只会使D数组的元素**D[L]+1**，元素**D[R+1]-1**，其它保持不变；
假设对A数组的区间**[L,R]整体-1**，那么只会使D数组的元素**D[L]-1**，元素**D[R+1]+1**，其它保持不变；

我们想要将D数组的[2,n]都调整为0
对于D数组，一次操作可以使“**D[i]+1同时D[j]-1**”,即i,j两个元素对消
也可以使“**D[1]+1同时D[i]-1**”或者 “**D[1]-1同时D[i]+1**”，即i与1对消。
也可以使“**D[n+1]+1同时D[i]-1**”或者 “**D[n+1]-1同时D[i]+1**”，即i与n+1对消。
也就是对于D[2..n]**一次操作可以同时改变两个元素的值，也可以改变一个元素的值**

所以对于第1问，只需计算D数组中所有**正数的总和Sum1**与所有**负数的总和Sum2**
答案所求的**最少操作次数=max(Sum1,|Sum2|)**

数列操作 解题分析

讨论第2问，求数列的方案数：

在第1问，已计算出了D数组中所有**正数的总和Sum1**与所有**负数的总和的绝对值Sum2**

将D[2...n]中的i,j配对消除所需次数为min{Sum1,Sum2}

此后D数组中还剩|Sum1-Sum2|个数没有变为0，现在可以与D[1]或D[n+1]配对消除操作次数为|Sum1-Sum2|，所以最终D[1]的取值可能有|Sum1-Sum2|+1种。

所以，最终第2问的答案为|Sum1-Sum2|+1

```
int main()
{
    ......
    for(i=n;i>1;i--)
        if(a[i]-a[i-1]>0)  Sum1+=a[i]-a[i-1];
        else   Sum2+=a[i-1]-a[i];
    cout<<max(Sum1,Sum2)<<endl
    cout<<abs(Sum1-Sum2)+1<<endl;
}
```

借教室 NKOJ1887

我们要处理接下来n天的借教室信息，其中第i天学校有ri个教室可供租借。共有m份订单，每份订单用三个正整数描述，分别为dj, sj, tj，表示某租借者需要从第sj天到第tj天租借教室（包括第sj天和第tj天），每天需要租借dj个教室。

对于每份订单，我们只需要每天提供dj个教室，而它们具体是哪些教室，每天是否是相同的教室则不用考虑。

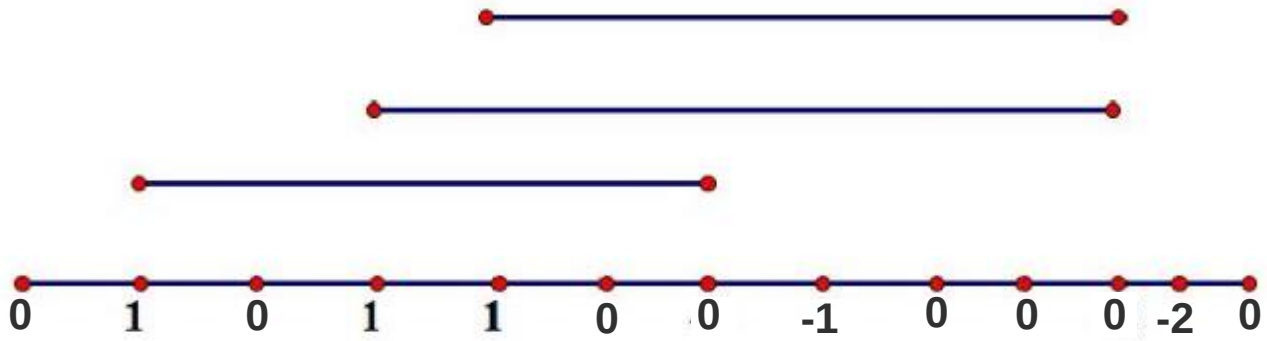
借教室的原则是先到先得，也就是说我们要按照订单的先后顺序依次为每份订单分配教室。如果在分配的过程中**一旦遇到一份订单无法完全满足，则需要停止所有教室的分配**，通知当前申请人退单。

“无法满足”指从第sj天到第tj天中有至少一天剩余的教室数量不足dj个。现在我们需要知道，是否会有订单无法完全满足。如果有，需要通知哪一个申请人。

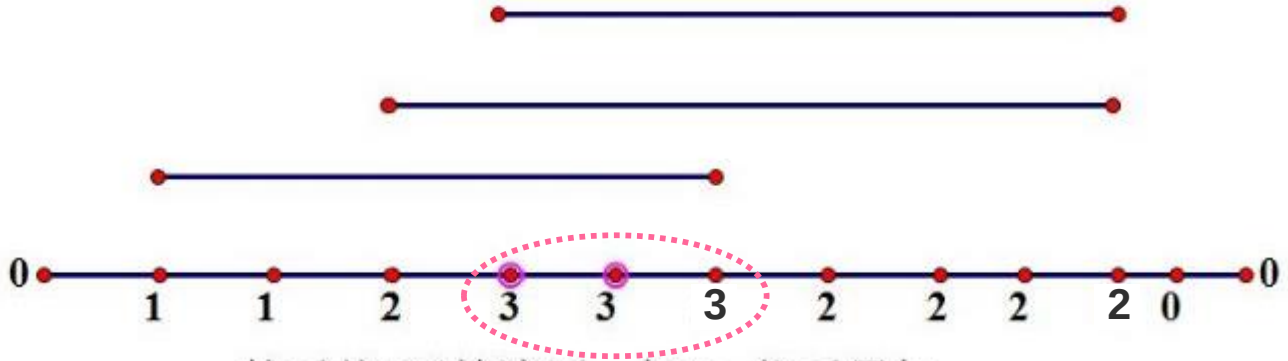
$1 \le n, m \le 10^6, 0 \le ri, dj \le 10^9, 1 \le sj \le tj \le n$

我们用**差分数组**来解决它！

技巧：求被所有线段都覆盖过的点？



可以考虑区间求交的方法：新建一个flag数组，区间开头+1，结尾-1



然后从1开始遍历，把flag往后累加，
如果当前点的flag等于区间的个数，
它就是所有区间交集中的一个点

借教室 解题分析

显然可以把每个要求订单看作是一条线段，然后每条线段都有厚度（需要的教室数量），叠加后判断是否超过可以承受的即可。

于是设置差分数组S[i]

对于一个订单[Li,Ri]区间每天需要Di间教室: S[Li]+Di，S[Ri+1]-Di

最后从左往右扫描一遍，第i天需要的教室数Need[i]=Need[i-1]+S[i]

若第i天可用的教室A[i]<Need[i],则无法满足！

具体实现时采用二分答案的方式，二分[1...mid]号订单是否满足，用差分数组去验证。

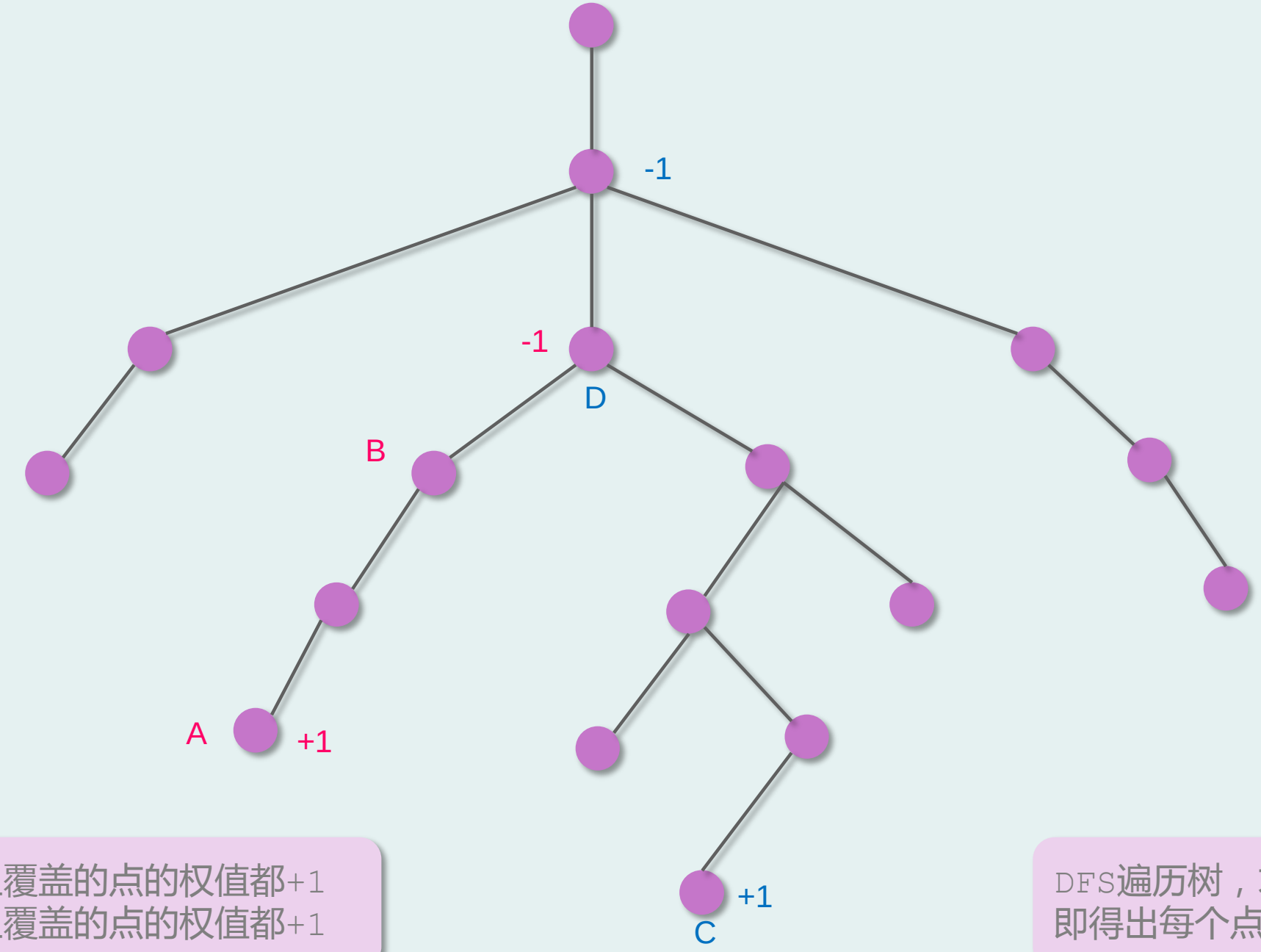
时间复杂度O(nlogn)

//参考代码

```
main()
{
   //二分答案
    int L=1,R=m;
    while (L<=R)
    {
        mid=(L+R)>>1;
        if (!judge(mid))ans=mid,R=mid-1;
        else L=mid+1;
    }
    if (!ans) printf("0\n");
    else printf("-1\n%d\n",ans);
}
```

```
bool judge(int mid)
{
    memset(b,0,sizeof(b));
    int Need=0;
    for (i=1; i<=mid; i++) //构造差分数组
    { S[L[i]]+=D[i]; S[R[i]+1]-=D[i];}
    for(i=1; i<=n; i++)
    {
        Need+=S[i];
        if (Need>A[i]) return false;
    }
    return true;
}
```

树上差分



将A到B路径上覆盖的点的权值都+1
将C到D路径上覆盖的点的权值都+1

DFS遍历树，求出树上前缀和。
即得出每个点的权值。

Max Flow NKOJ3605

给定一棵有N个点的树，所有节点的权值初始时都为0。

有K次操作，每次指定两个点s,t，将s到t路径上所有点的权值都+1。

请输出K次操作完毕后权值最大的那个点的权值。

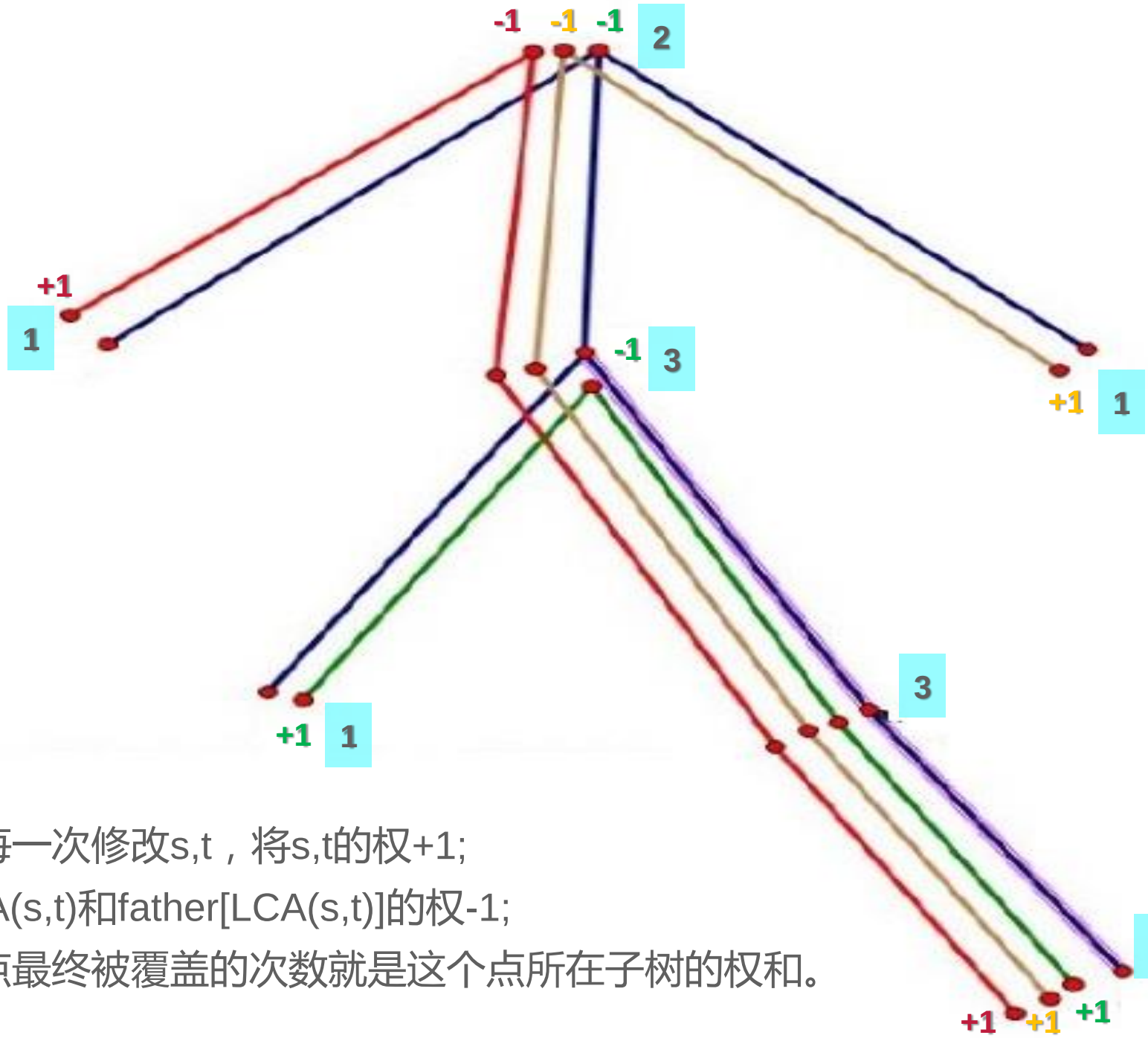
$2 \le N \le 50,000$　　$1 \le K \le 100,000$

对于每一次修改s,t，

1.将s,t的权+1;

2.将LCA(s,t)和father[LCA(s,t)]的权-1;

一个点最终被覆盖的次数就是这个点所在子树的权和。

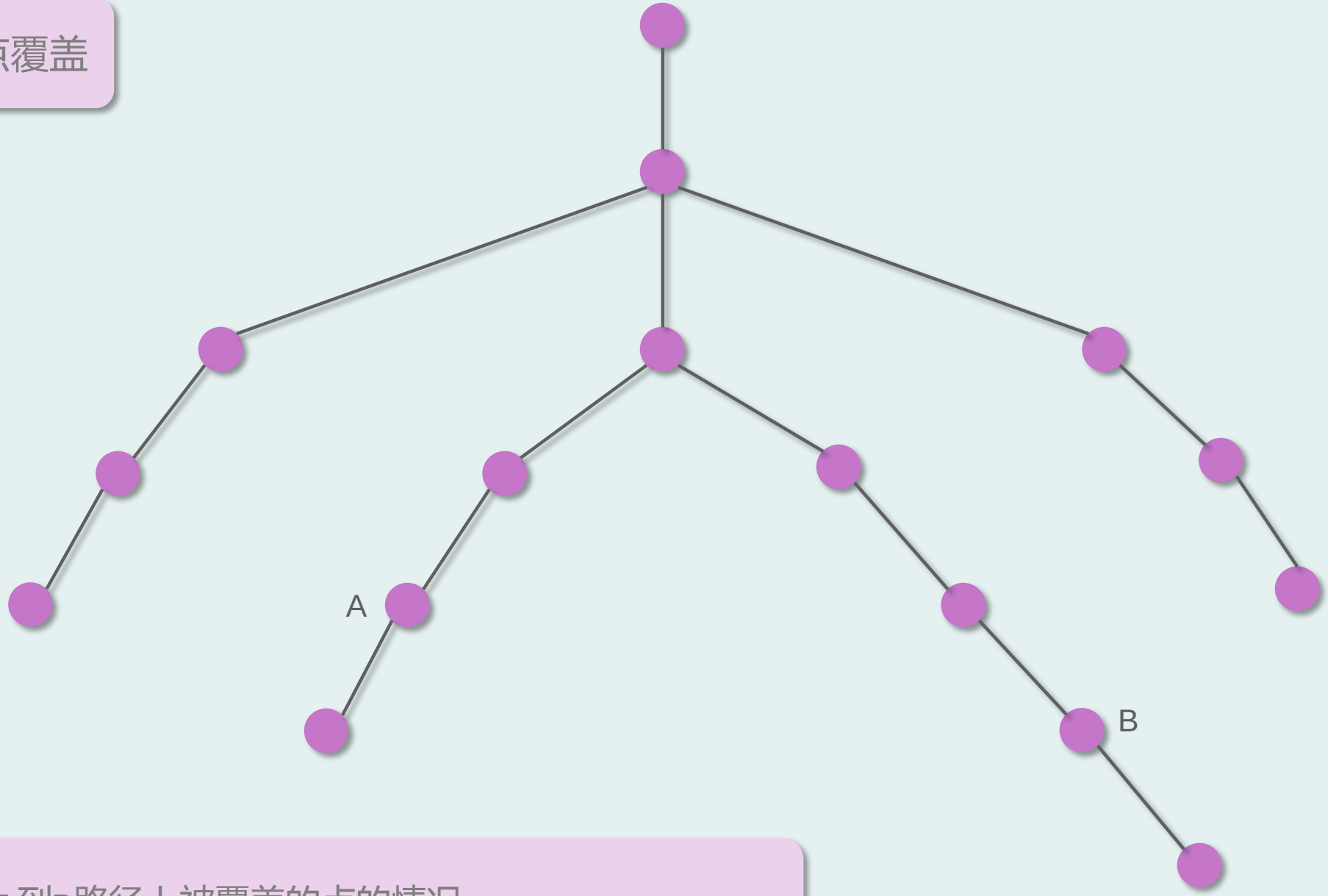


对于每一次修改s,t，将s,t的权+1;

将LCA(s,t)和father[LCA(s,t)]的权-1;

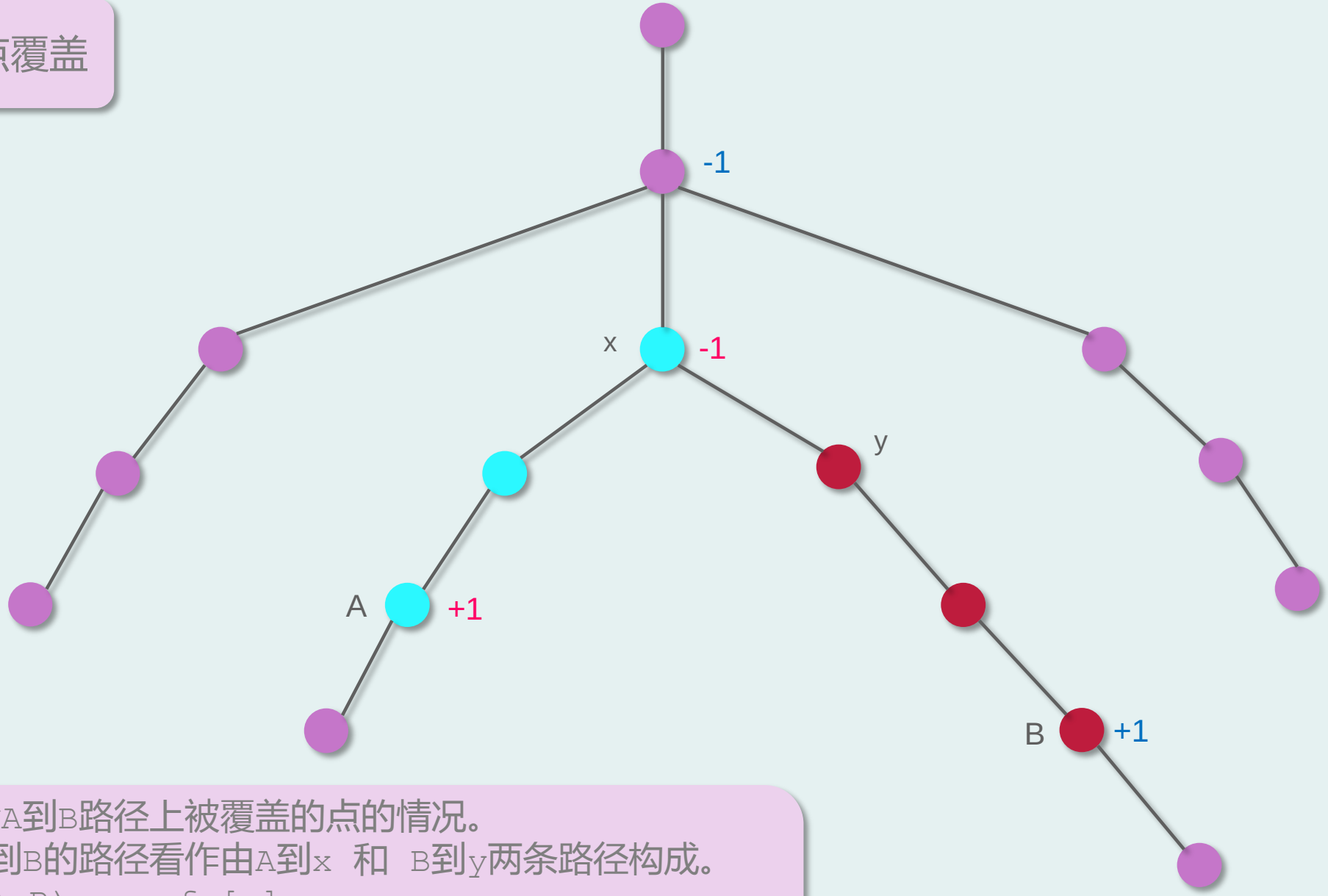
一个点最终被覆盖的次数就是这个点所在子树的权和。

路径的点覆盖



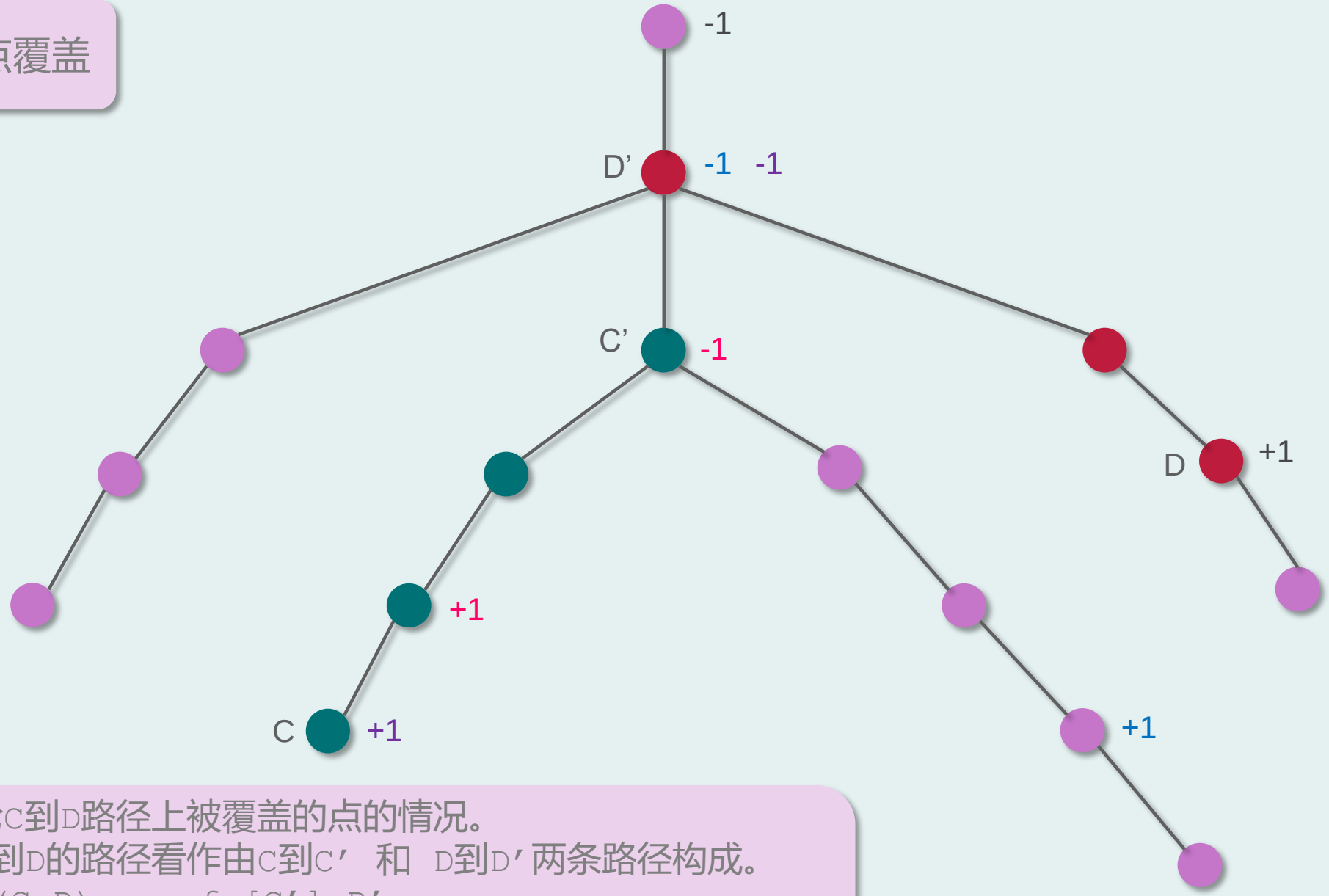
我们讨论A到B路径上被覆盖的点的情况。

路径的点覆盖



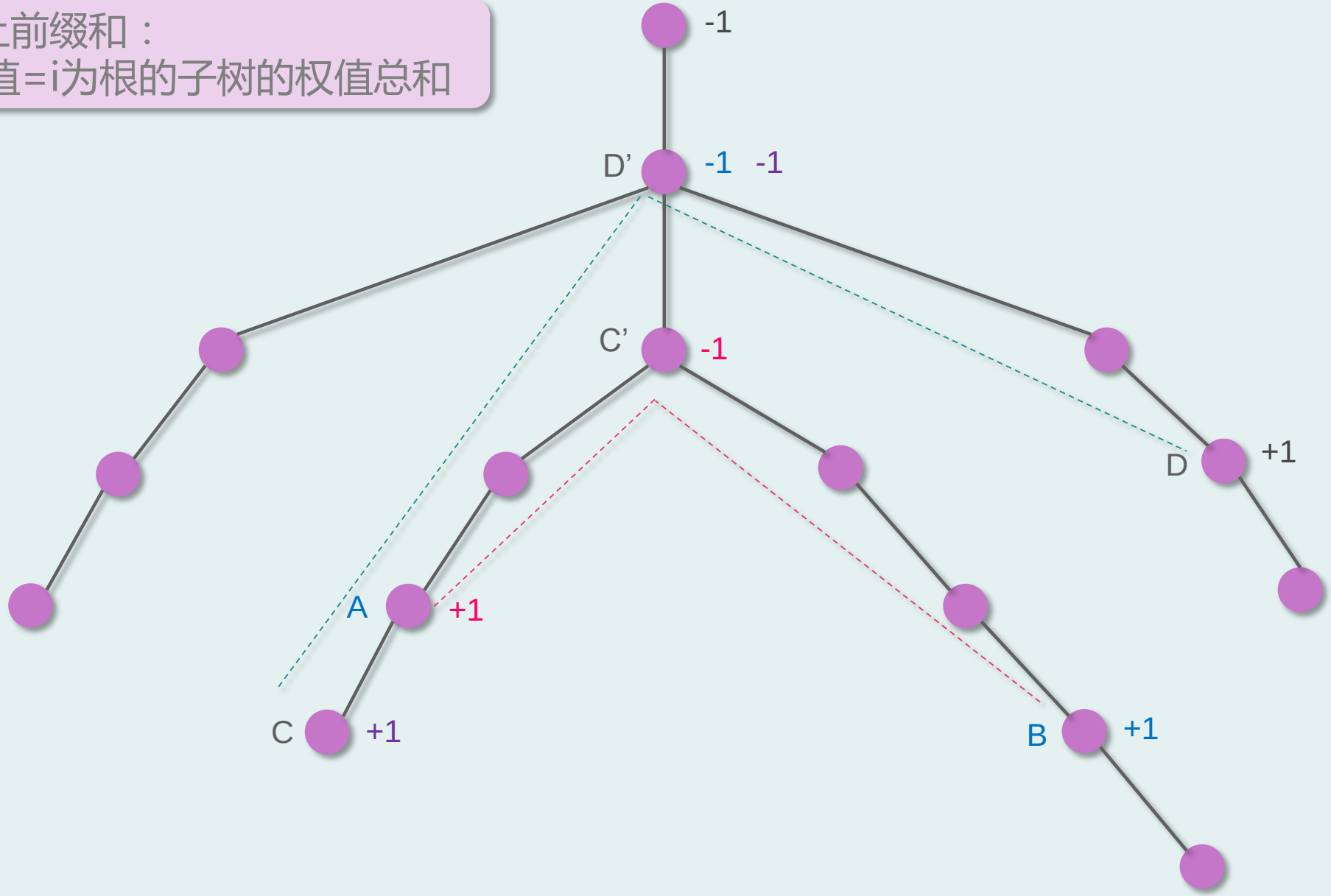
我们讨论A到B路径上被覆盖的点的情况。
可以把A到B的路径看作由A到x 和 B到y两条路径构成。
`x=Lca(A,B)` `fa[y]=x`
对于A到x，根据差分操作有`Cnt[A]++, Cnt[fa[x]]--`
对于B到y，根据差分操作有`Cnt[B]++, Cnt[fa[y]]--`

路径的点覆盖

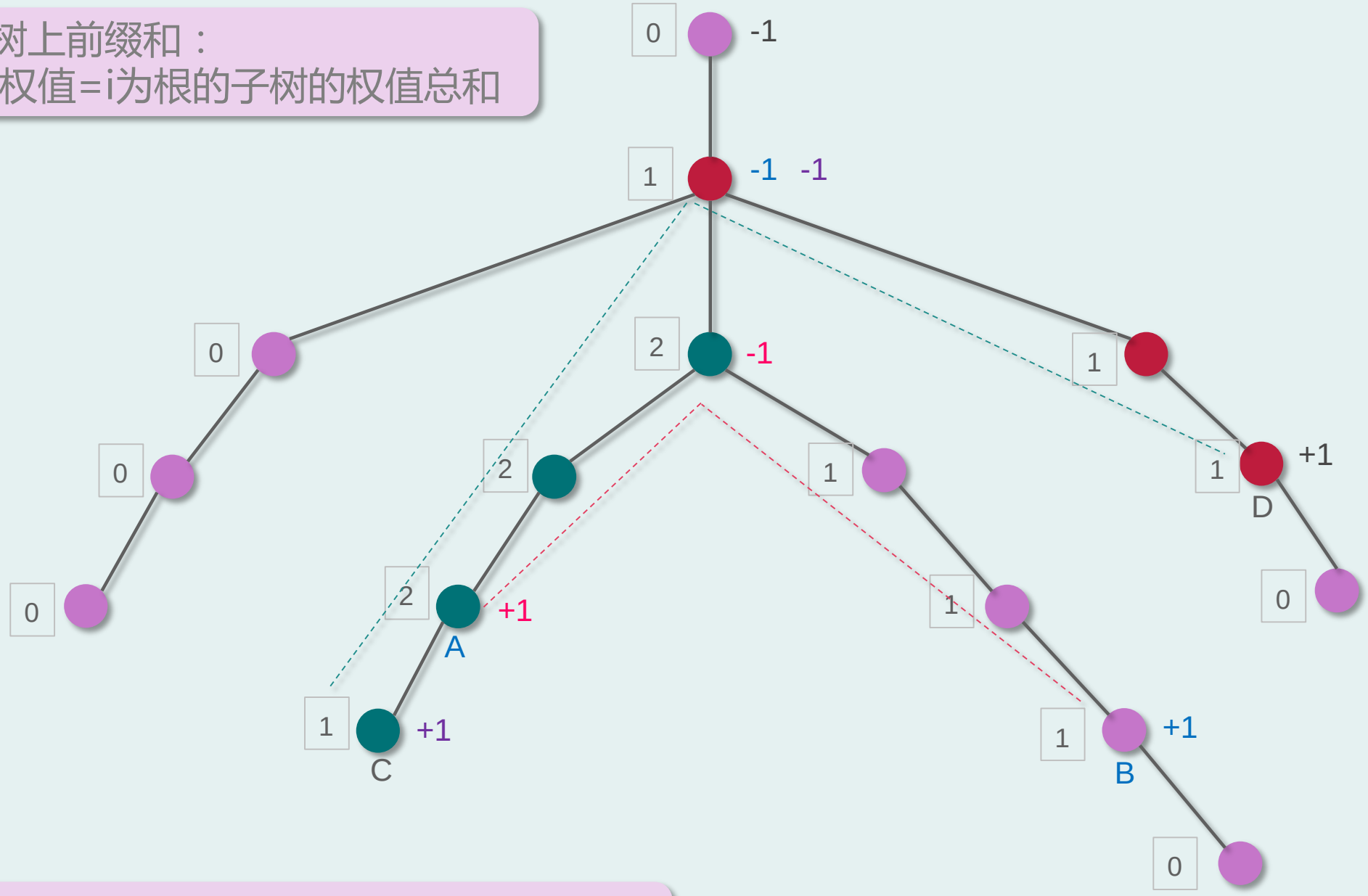


我们讨论C到D路径上被覆盖的点的情况。
可以把C到D的路径看作由C到C′ 和 D到D′两条路径构成。
D′=Lca(C,D)　　fa[C′]=D′
对于C到C′，根据差分操作有Cnt[C]++, Cnt[fa[C′]]--
对于D到D′，根据差分操作有Cnt[D]++, Cnt[fa[D′]]--

统计树上前缀和：
点i的权值=i为根的子树的权值总和



统计树上前缀和：
点i的权值=i为根的子树的权值总和



每个点的权值代表了该点被路径覆盖的次数

树上差分的结论1：询问树上**每个点**被覆盖的次数

1.对于每条路径：

a.在树中将所有路径起、止点权值加1；

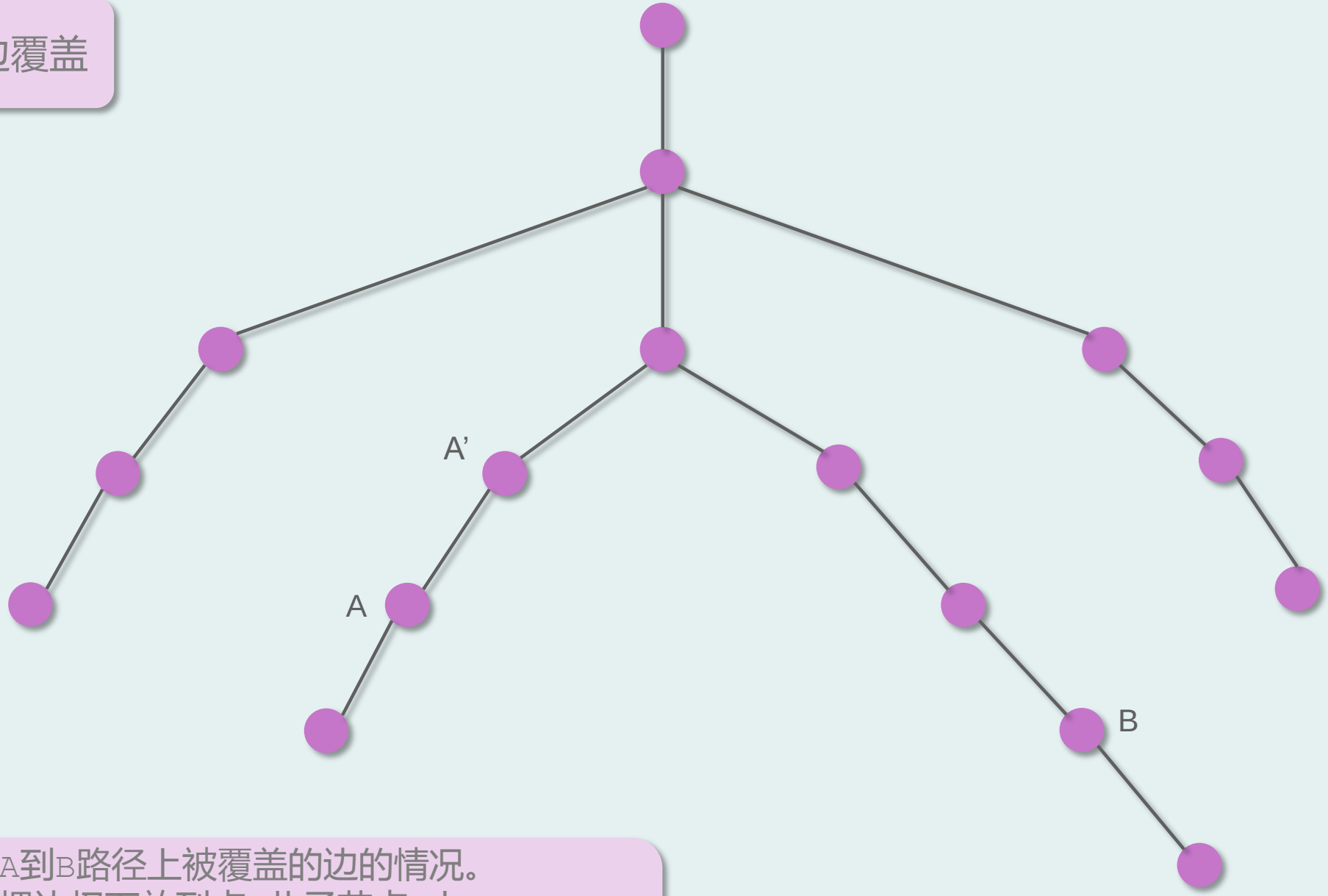
b.起、止点的lca权值减1，lca的父亲节点的权值减1；

2.点被覆盖的次数：

a.从所有叶节点开始把权值往上累加；

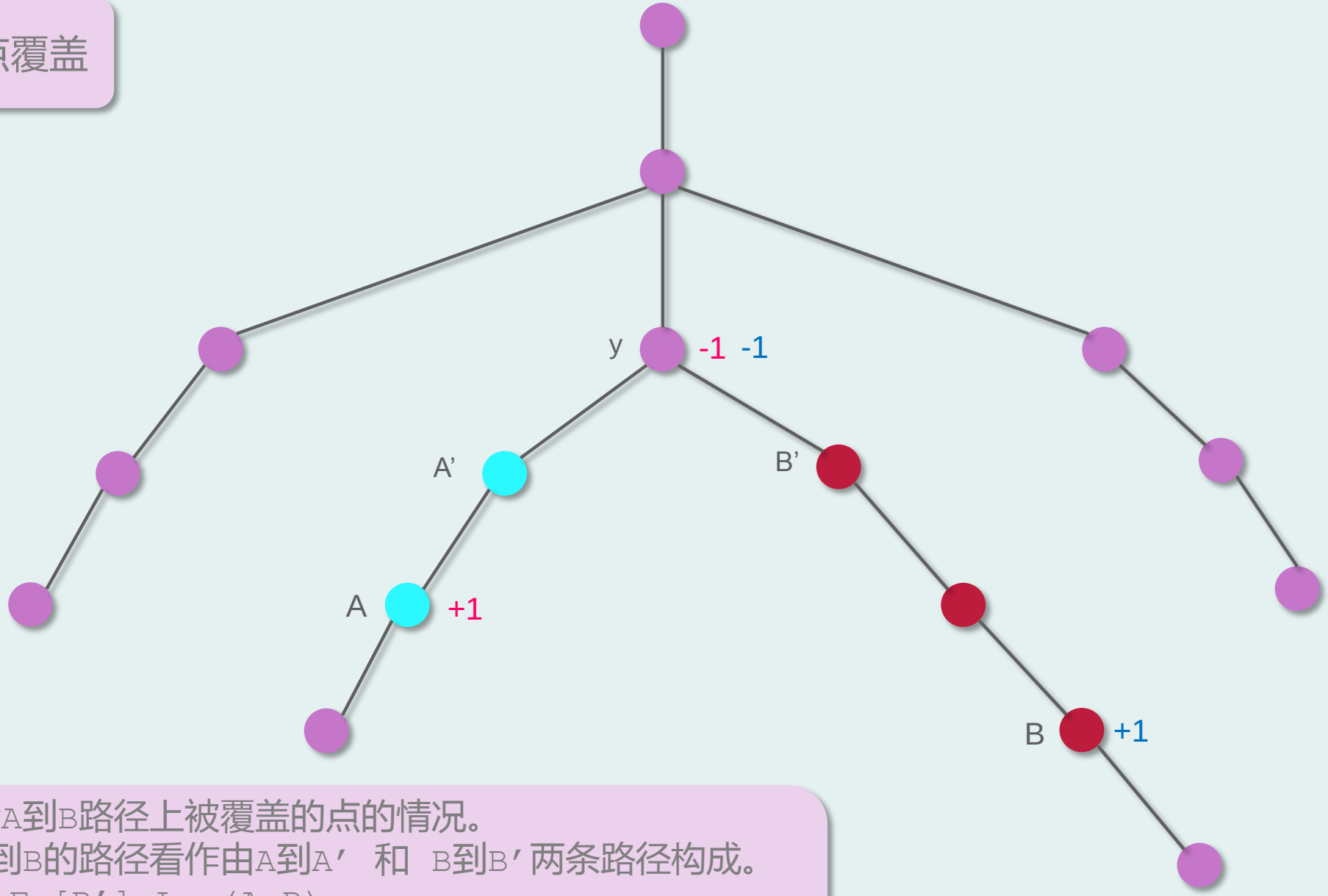
b.每个点的最终权值为该点被覆盖的次数；

路径的边覆盖



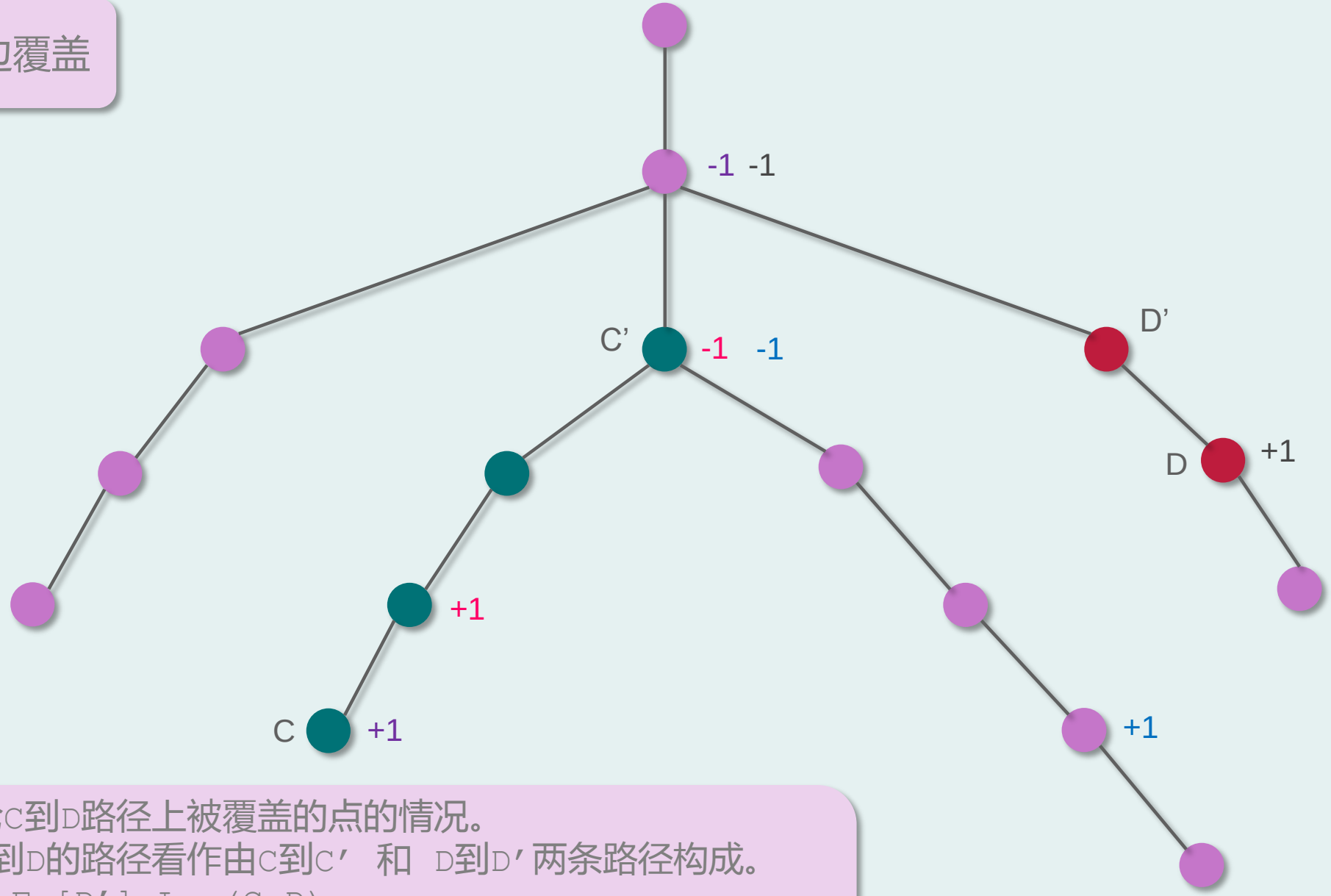
我们讨论A到B路径上被覆盖的边的情况。
我们一般把边权下放到点（儿子节点）上。
比如A’—— A这条边的边权记再点A上。
覆盖了A就等价于覆盖了A—A’这条边

路径的点覆盖



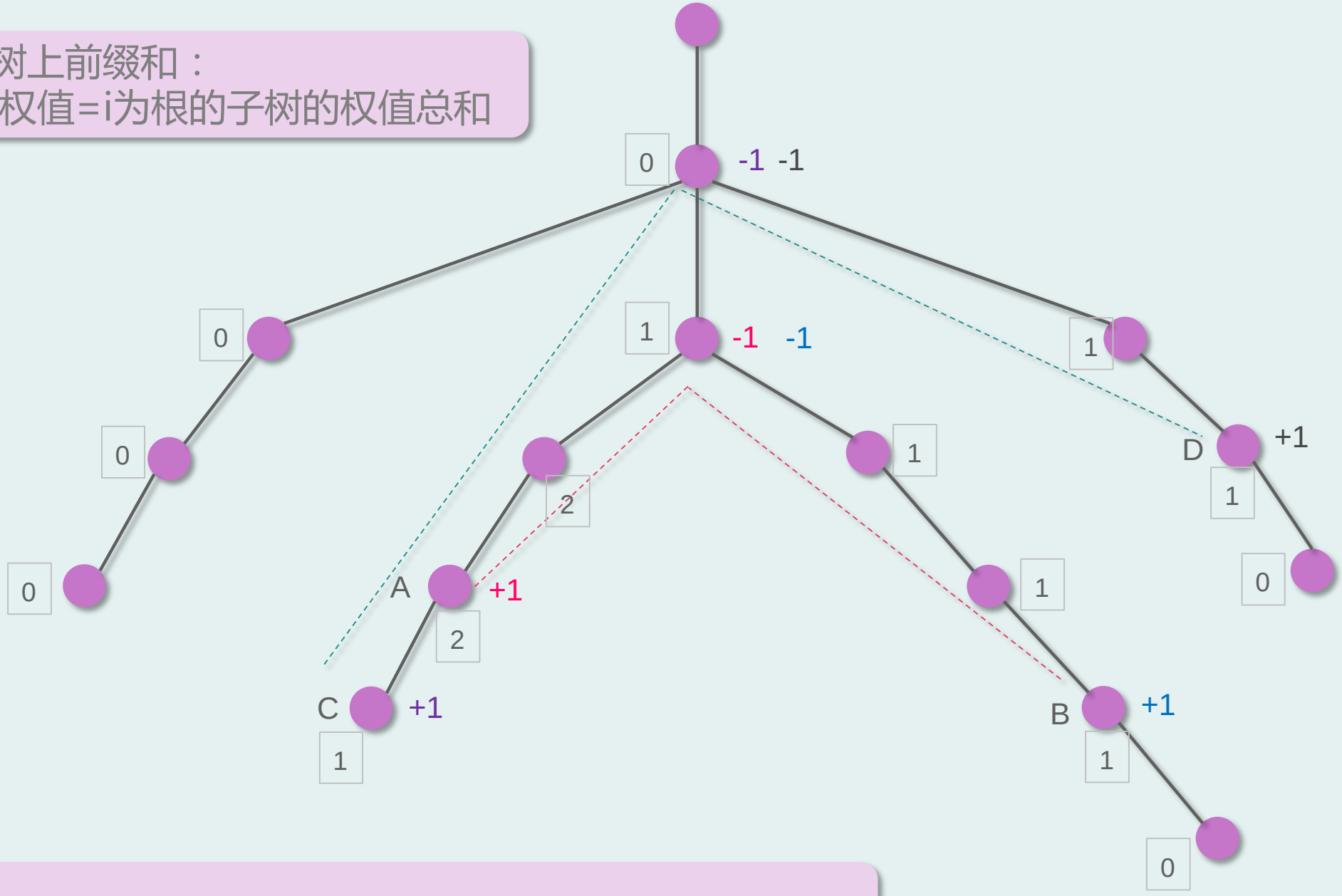
我们讨论A到B路径上被覆盖的点的情况。
可以把A到B的路径看作由A到A’ 和 B到B’两条路径构成。
Fa[A’]=Fa[B’]=Lca(A,B)
对于A到A’，根据差分操作有Cnt[A]++, Cnt[fa[A’]]--
对于B到B’，根据差分操作有Cnt[B]++, Cnt[fa[B’]]--

路径的边覆盖



我们讨论C到D路径上被覆盖的点的情况。
可以把C到D的路径看作由C到C’ 和 D到D’两条路径构成。
Fa[C’]=Fa[D’]=Lca(C,D)
对于C到C’，根据差分操作有Cnt[C]++, Cnt[fa[C’]]--
对于D到D’，根据差分操作有Cnt[D]++, Cnt[fa[D’]]--

统计树上前缀和：
点i的权值=i为根的子树的权值总和



每个点的权值代表了该点表示的边被路径覆盖的次数

树上差分的结论2：询问树上**每条边**被覆盖的次数：

1.对于每条路径：

a.在树中将所有路径起、止点权值加1；

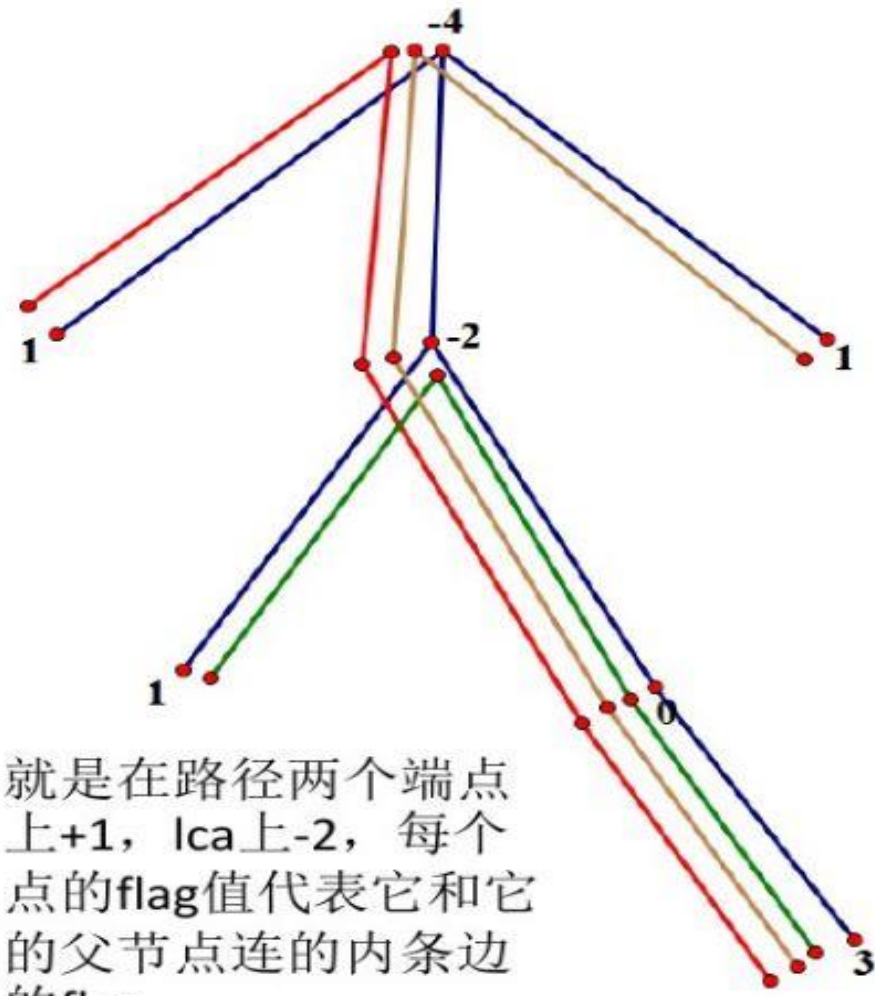
b.起、止点的lca权值减2；

2.边被覆盖的次数：

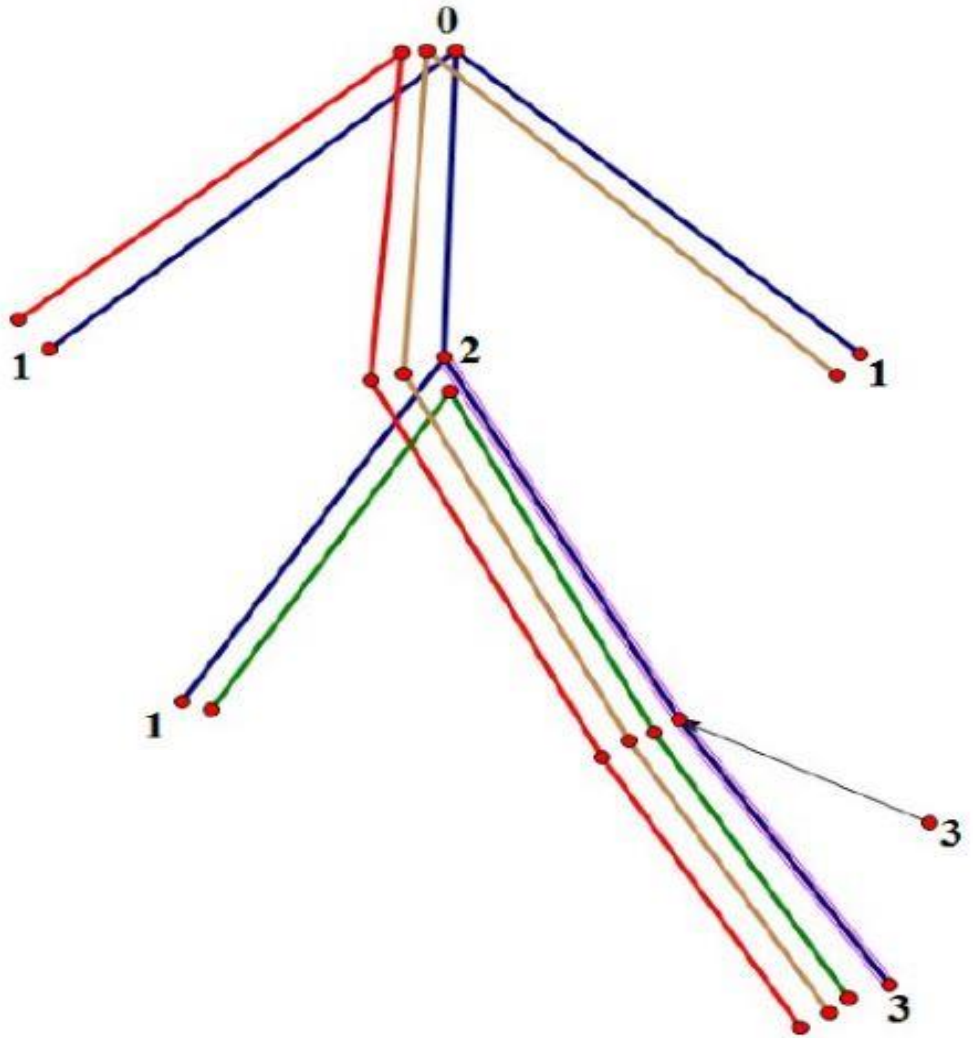
a.从所有叶节点开始把权值往上累加；

b.每个点的最终权值为该点与它父亲所连这条边被覆盖的次数；

延伸到树上



就是在路径两个端点上+1，lca上-2，每个点的flag值代表它和它的父节点连的内条边的flag



【NOIP2015 Day2】运输计划 NKOJ3569

有n个星球，有n-1条双向航道，每条航道建立在两个星球之间，这n-1 条航道连通了所有星球。

小P的公司有m个运输计划，每个运输计划的起点和终点可能不同，每个运输计划形如：有一艘物流飞船从Ui号星球沿最快的路径飞行到Vi号星球去。对于航道j，任意飞船驶过它所花费的时间为Tj。

允许小P把某一条航道改造成虫洞，飞船驶过虫洞不消耗时间。在虫洞建设完成后，m个运输计划会同时开始，所有飞船一起出发。小P可以自由选择将哪一条航道改造成虫洞， 求出小P的公司完成所有运输计划所需要的最短时间是多少？

1<=n,m<=300000

时空限制：2s,256m

中心思想：给定一棵树和一些点对，可以将一条树边的权值变为0，使点对最大距离最小。

【NOIP2015 Day2】运输计划 NKOJ3569

本题要求：m条线路中，使最长那一条的路线尽可能短。有明显的”二分答案”信号。

问题1：怎样快速求出树中任意两点x到y的距离？

预处理：通过一次DFS求出每个点到根的距离Dis[ ]

点x到y的距离：Len[x,y]= **Dis[x]+Dis[y]-2*Dis[LCA(x,y)]**

问题2：二分答案Ans的上下界是多少？

下界：0

上界：m个点对中，最远那一对点的距离，即 Max{Len[x,y]}

【NOIP2015 Day2】运输计划 NKOJ3569

问题3：怎样验证二分的答案Ans是否可行？

改造某条航道后，所有m个点对的距离都<=Ans,即可行

问题4：改造哪一条航道？

设M条路线中，长度>Ans有k条

要改造的那条航道一定存在于这k条线路的每一条中。即改造后，k条线路长度都会减少。

我们要找出这k条线路中，所有被经过了k次的边。改造其中最长的一条，然后统计改造后，这k条路线长度是否都<=Ans。该边的长度>=Max{Len[x,y]}-Ans

统计m条线路中，每条边被覆盖的次数—树上差分。

对于点x到点y的路径，Cnt[x]++,Cnt[y]++,Cnt[LCA(x,y)]-=2

通过从根出发求一次树上前缀和，统计每条边被覆盖的次数。

找出覆盖次数为k的边中，是否存在长度>= Max{Len[x,y]}-Ans，存在即可行

树上差分，典型例题：

NKOJ 3560，3605，4314，1888，4238

其他题目：

NKOJ 3782