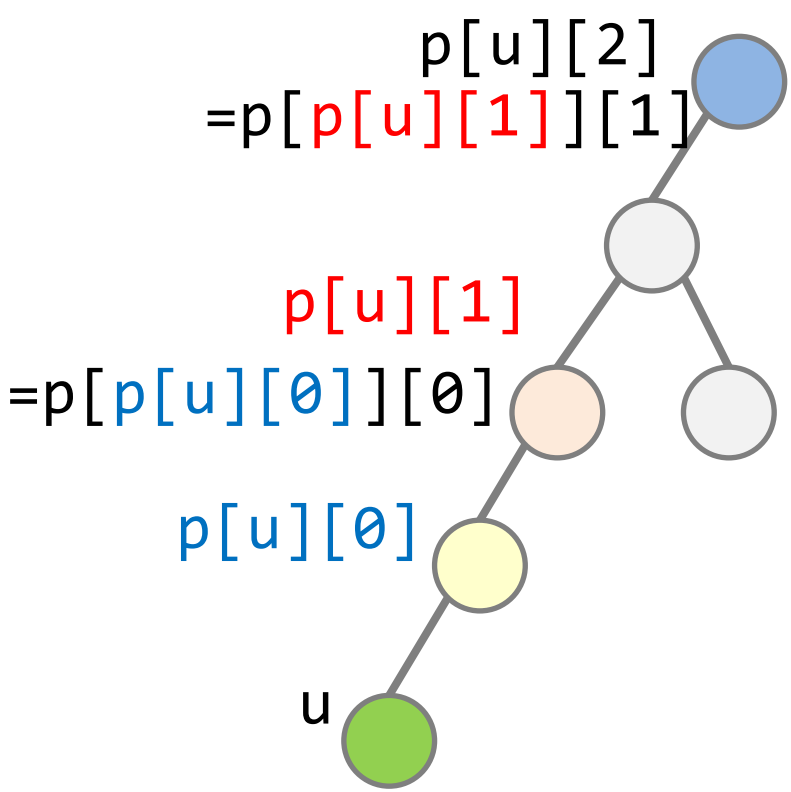
C++算法

快快编程1839

LA(u,k)

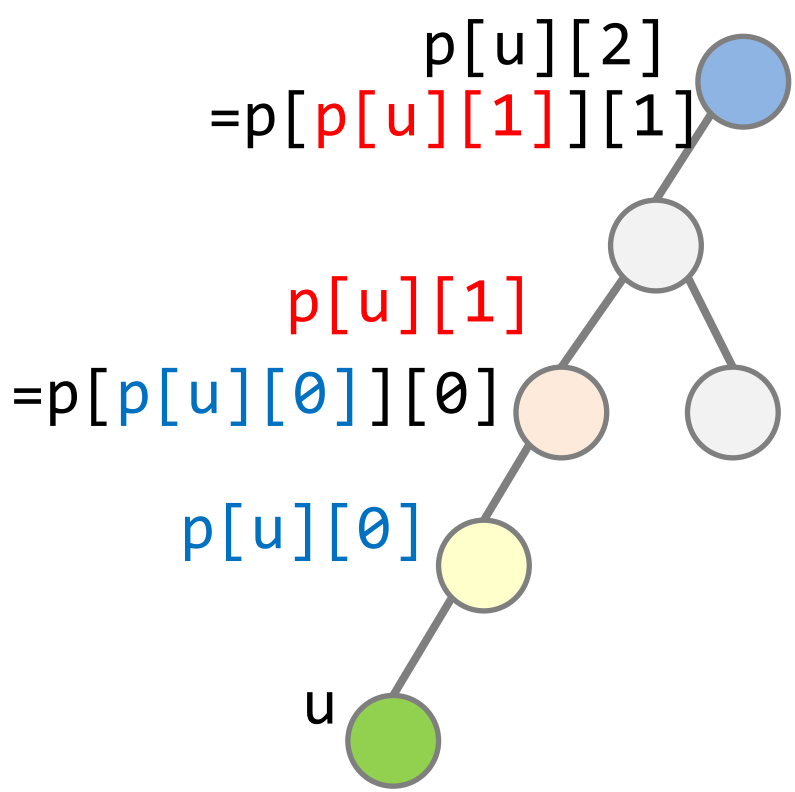
暴力	对于随机数据 单次查询平均复杂度O(logn)
简化链状	链状用一维数组储存节点序列 数组能快速索引
	树链剖分
倍增 binary lifting	预处理ST表(稀疏表) p[u][i]代表u的第2^i个祖先



请写出p[u][i]的递推式

L=log(n)/log(2)+1;

```
3  const int N=200009;
4  vector<int> to[N];
5  int L,n,q,p[N][20];
6  void dfs(int u,int fa){
7      p[u][0]=fa;
8      for(int i=1;i<=L;++i)
9          p[u][i]=p[p[u][i-1]][i-1];
10     for(int i=0;i<to[u].size();++i)
11         if(to[u][i]!=fa) dfs(to[u][i],u);
12 }
```



```
13 int LA(int u,int k){
14
15
16
17
18 }
```

$k=(7)_{10}=(111)_2$
(1<<0)&k非0 (1<<1)&k非0 (1<<2)&k非0

```
60      scanf("%d",&m);
61      for(int i=1;i<=m;++i){
62          int v,k;
63          scanf("%d %d",&v,&k);
64          if(d[v]<=k)continue;
65          int u=LA(v,k);
66          q[u].push_back((query){i,k});
67      }
```

```
26 void aNode(int u,bool tag){
27     if(tag)++cnt[d[u]];
28     else
29 }
30 void aTree(int u,bool tag){
31     for(int i=tI[u];i<=tO[u];++i)
32         aNode(id[i],tag);
33 }
```

```
34 void dfs(int u,bool hvy){
35     for(int i=hd[u];i;i=nxt[i])
36         if(son[u]!=to[i])dfs(to[i],0);
37     if(son[u])dfs(son[u],1);
38     for(int i=hd[u];i;i=nxt[i])
39         if(son[u]!=to[i])aTree(to[i],1);
40     aNode(u,1);
41     for(int i=0;i<q[u].size();++i)
42         ans[q[u][i].id]=
43     if(hvy)return;
44     aTree(u,0);
45 }
```

快快编程1872

cntAll[c]表示整棵树里c号颜色出现次数

cnt[c]表示子树内c号颜色出现次数

若cnt[c]==cntAll[c]则c号颜色
全部在子树内不在子树外,对答案贡献为0

若cnt[c]==0则c号颜色
全部在子树外不在子树内,对答案贡献为0

若0<cnt[c]<cntAll[c]则c号颜色
对答案贡献为1

快快编程1840

有根树上n个节点
节点u的字母为ltr[u]
共q个问询：
节点u子树内深度恰为h的节点字母
全部用上能否重组成回文串

给定字母集合
能重组成回文串的条件是什么？

出现奇数次的字母最多1个时
字母集合可以重组成回文串

```
27 void aNode(int u){
28     int&d=::d[u];
29     int&ltr=::ltr[u];
30     cOdd[d]-=cnt[d][ltr];
31     cnt[d][ltr]^=1;
32     cOdd[d]+=cnt[d][ltr];
33 }
34 void aTree(int u){
35     for(int i=tI[u];i<=tO[u];++i)
36         aNode(id[i]);
37 }
```

应用场景

小并大/启发式合并

子树信息

离线问询

路径问询能做吗？

快快编程1873

变种：将点权改成边权
对比快快1808

哪一题难为什么

变种：将长度定义为点数
求长度恰为k的路径数

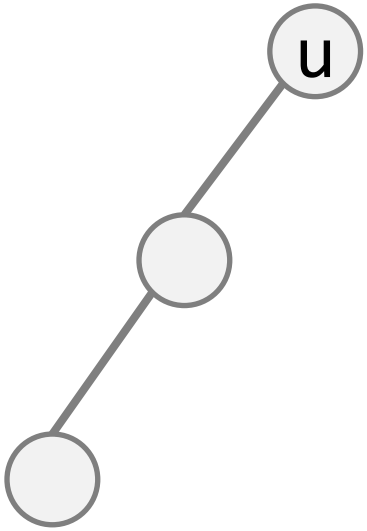
路径边权异或值	path[u]表示根到u路径内边权异或值
	dst(u,v)=path[u]^path[v]

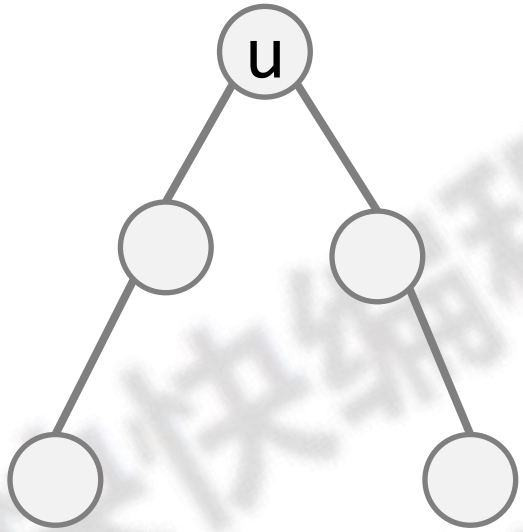
路径点权异或值	path[u]表示根到u路径内点权异或值
	dst(u,v)=path[u]^path[v]^val[lca(u,v)]
	val[u]为u的点权

路径点数	d[u]表示根到u路径内点数
	dst(u,v)=d[u]+d[v]-2*d[lca(u,v)]

如何将路径计数所需信息
转换为子树内信息统计

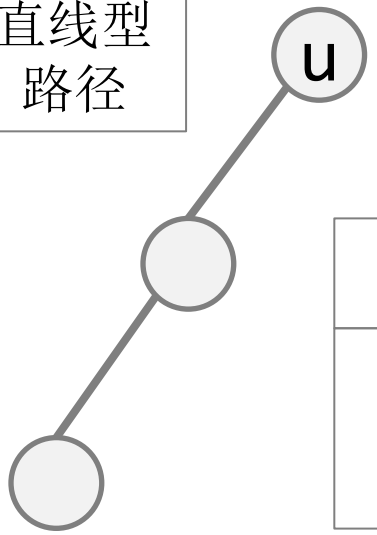
2种路径：直线，折线





都属于某棵子树内部

直线型
路径



快快1873

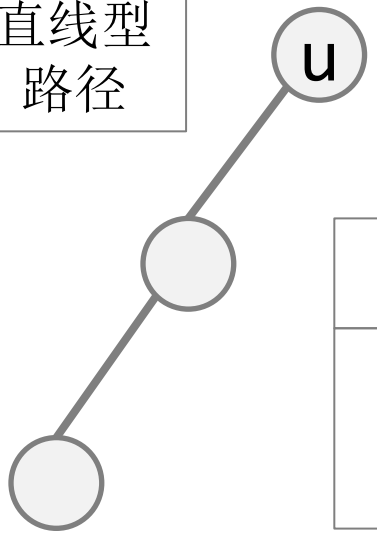
有多少条直线型路径上点编号异或值为0

有多少条直线型路径上点编号异或值为0
且以u为最高点

路径1端已定,计算符合条件的另1端的个数

用三脚架
形态思考

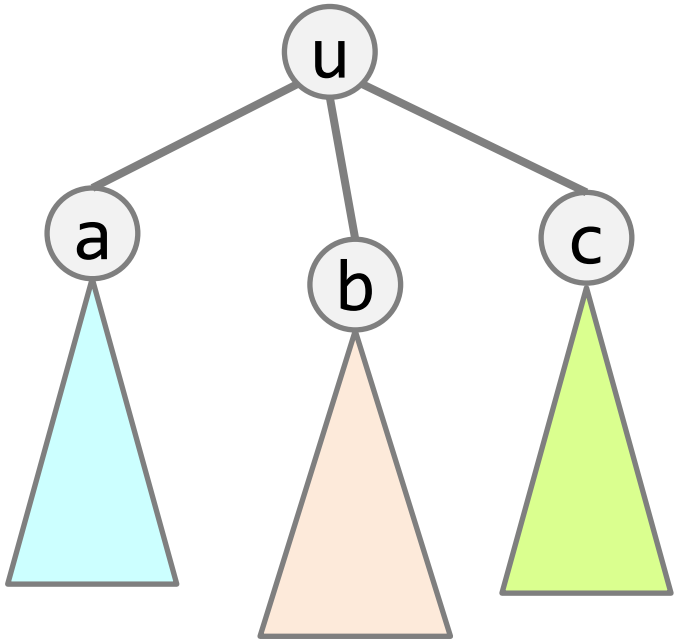
直线型路径



快快1873

有多少条直线型路径上点编号异或值为0

有多少条直线型路径上点编号异或值为0
且以u为最高点



cnt[p]记录u儿子子树a,b,c子树内某点到根异或为p的路径条数

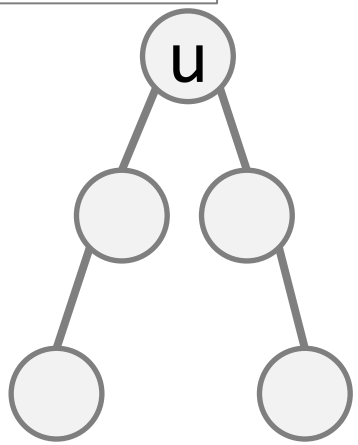
x^path[u]^u==0
即x==path[u]^u

路径数：cnt[path[u]^u]

若题意改成求异或值为k的路径数

折线型
路径

快快1873



有多少条折线型路径上点编号异或值为0

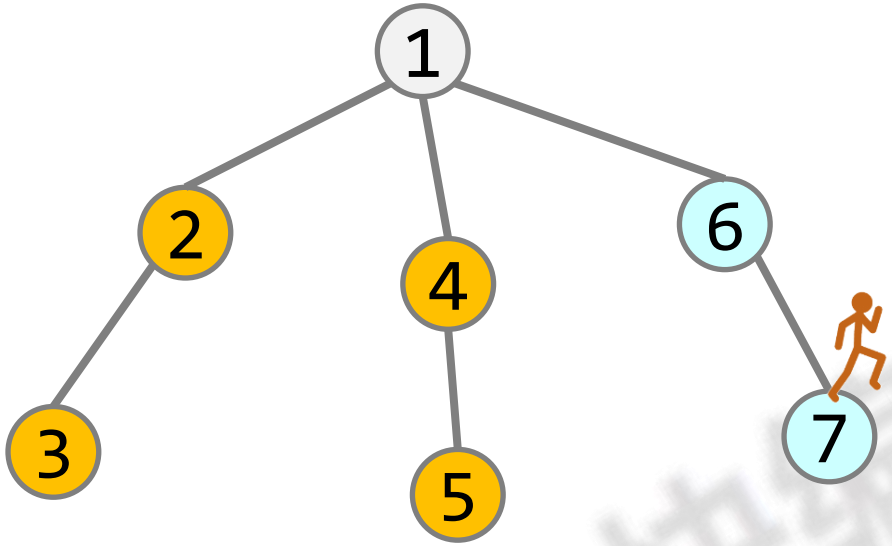
有多少条折线型路径上点编号异或值为0
且是以u为lca的折线型

对于通过子树根u的路径
如何设计枚举过程

枚举1端算符合条件的另1端的个数

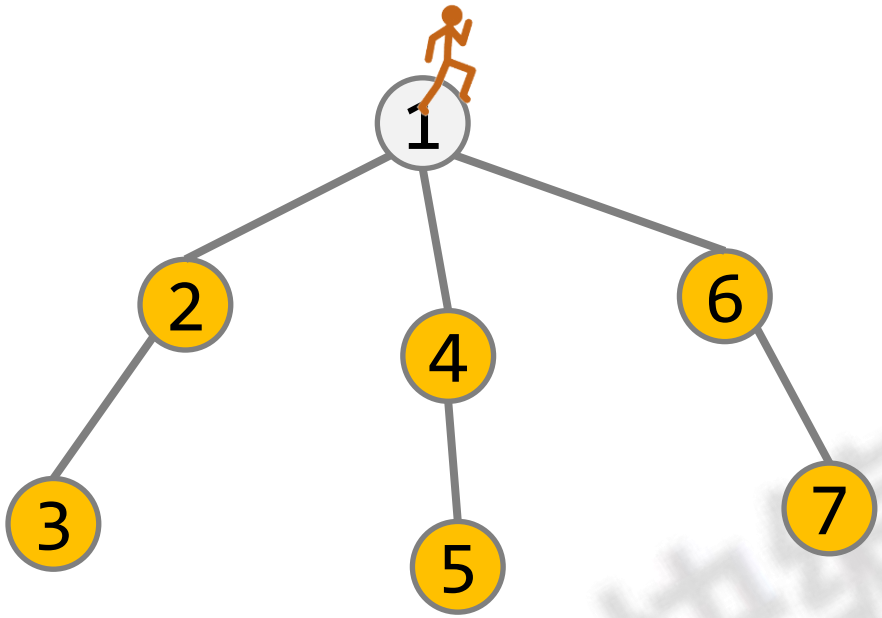
用三脚架
形态思考

通过子树根1的路径
异或值为0的折线型路径数量



蓝配黄形成路径

通过子树根1的路径
异或值为0的折线型路径数量



蓝变黄

```cpp
void dfs(int u,int fa,bool hvy){
    for(int i=hd[u];i;i=nxt[i]){
        int v=to[i];
        if(v==son[u]||v==fa)continue;
        dfs(v,u,0);
    }
    if(son[u])dfs(son[u],u,1);
    res=0;
    for(int i=hd[u];i;i=nxt[i]){
        int v=to[i];
        if(v==son[u]||v==fa)continue;
        useTree(v,u);   addTree(v,1);
    }
    useNode(u,u);   addNode(u,1);
    ans[u]=res;
    if(hvy)return;
    addTree(u,0);
}
```

```
26 void useNode(int u,int lca){
27     res+=
28 }
29 void useTree(int u,int lca){
30     for(int i=tI[u];i<=tO[u];++i)
31         useNode(id[i],lca);
32 }
33 void addNode(int u,bool ADD){
34     if(ADD) ++cnt[path[u]];
35     else    --cnt[path[u]];
36 }
37 void addTree(int u,bool ADD){
38     for(int i=tI[u];i<=tO[u];++i)
39         addNode(id[i],ADD);
40 }
```

快快编程1841

出现奇数次的字母最多1个时
字母集合可以重组成回文串

path[u]用二进制压缩表示u到根
路径上各字母次数的偶奇01串

u到v路径上各字母次数的偶奇01串
path[u]^path[v]

可重组回文：01串中最多1个1

子树内路径不一定通过子树根

对于通过子树根u的路径
如何设计枚举过程

用三脚架形态思考

有根树上n个节点
每条边有字母，在a到v之间
对每个节点u问询：u子树中，
可重组回文路径最长有多长?

用x二进制压缩表示u到根
路径上各字母次数的偶奇01串

f[x]表示当前存档的点(黄色)中
到根路径01模式为x的点的最大深度

f[path[u]]表示当前存档的点(黄色)中
到根路径01模式为path[u]的点v的最大深度

该点v能和u搭配出可重组回文路径

```
42 void dfs(int u,bool hvy){
43     for(int i=hd[u];i;i=nxt[i])
44         if(son[u]!=e[i].v)
45             dfs(e[i].v,0),upd(ans[u],ans[e[i].v]);
46     if(son[u])dfs(son[u],1),upd(ans[u],ans[son[u]]);
47     for(int i=hd[u];i;i=nxt[i])
48         if(son[u]!=e[i].v)
49             useTree(e[i].v,u),addTree(e[i].v,1);
50     useNode(u,u), addNode(u,1);
51     upd(ans[u],bst);
52     if(hvy)return;
53     addTree(u,0);
54     bst=0;
55 }
```

快快编程1842

d[v]表示v深度:根到v节点数
dst[v]表示根到v路径长度

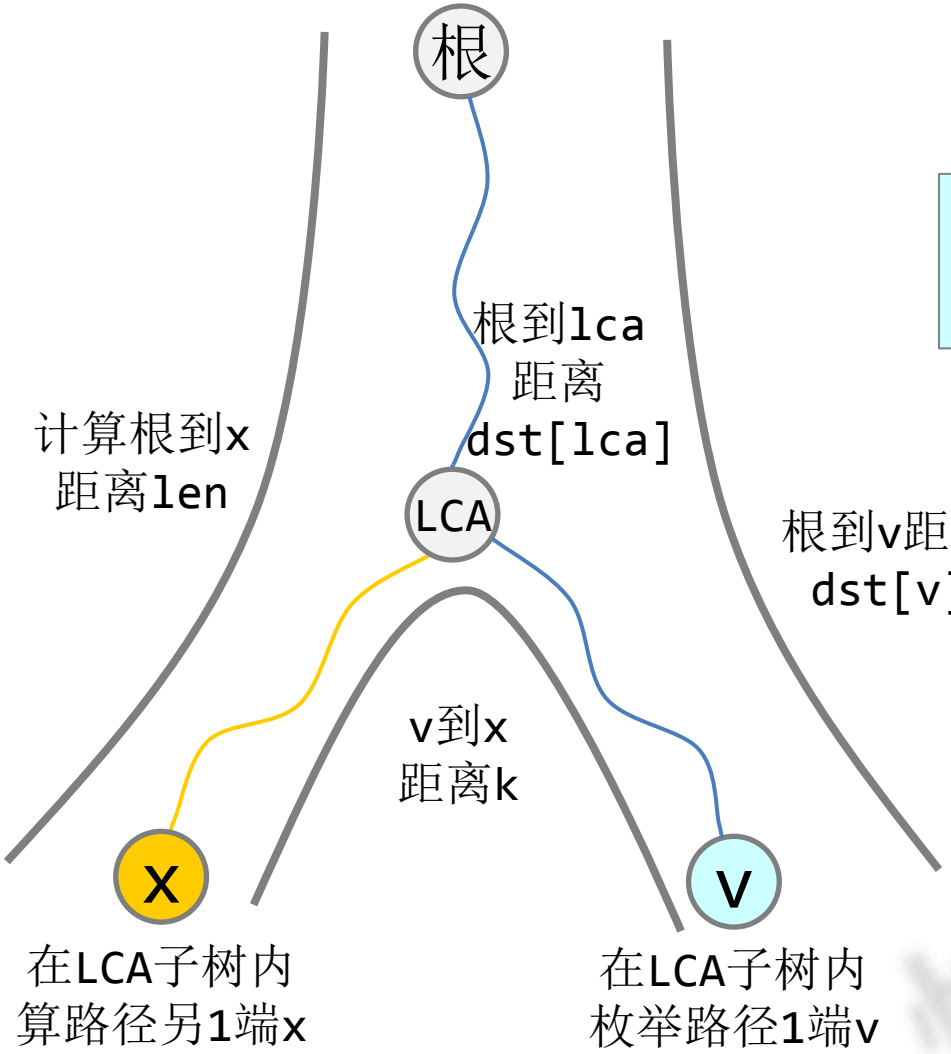
map<ll,int> f;
map<根到点距离,最少点数> f;

f[len]表示只考虑目前子树内部已加入部分里点集
某点v到原根的绝对距离为len的路径
至少用几个节点
即满足dst[v]==len时对应最小的d[v]

如何将路径计数所需信息
转换为子树内信息统计

2种路径：直线，折线

枚举1端算符合条件的另1端的个数

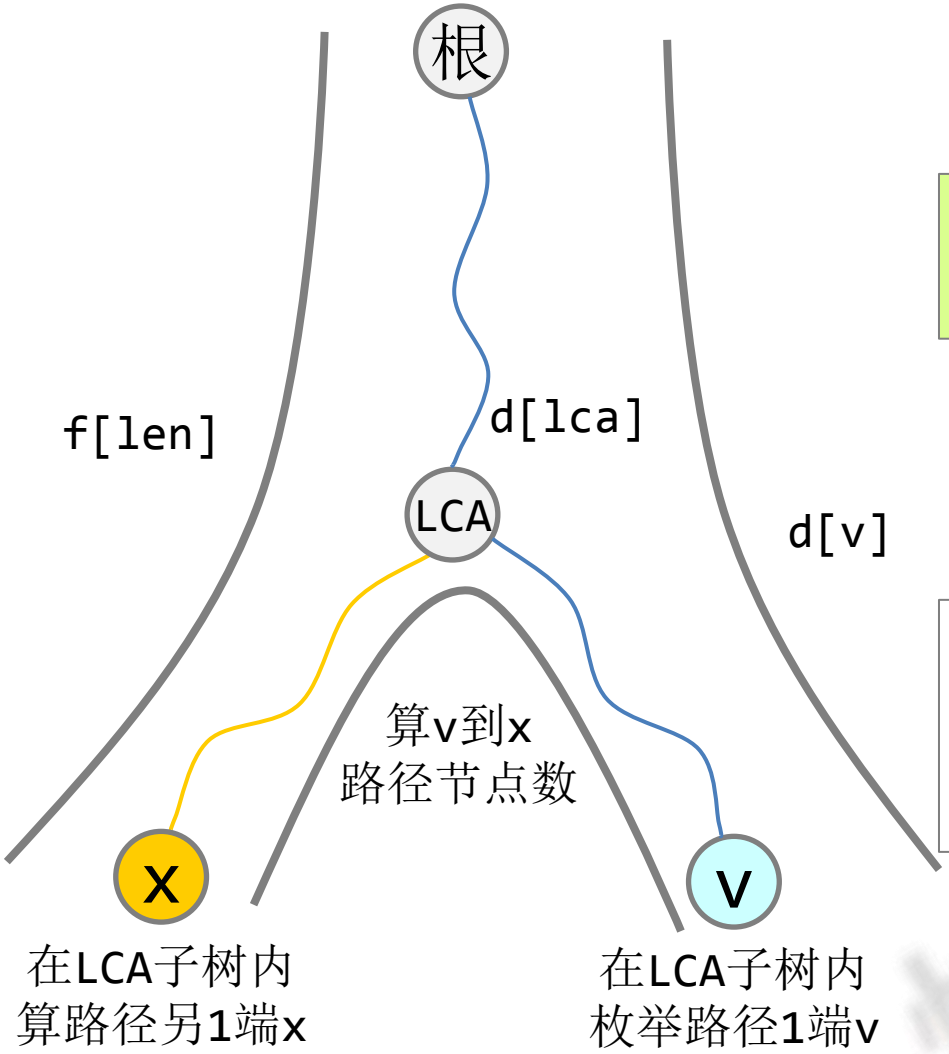


len=k-(dst[v]-dst[lca])+dst[lca]
=k-dst[v]+2*dst[lca]

易错点是什么

k<dst[v]-dst[lca]

即len<dst[lca]



当前从v出发长度k路径最少边数
=f[len]+d[u]-d[lca]*2

更新当前答案

ans[u]记录u子树内答案：
在u子树内部
距离为k的路径最少几条边

快快编程作业

1839,1840,1841,1842,1872,1873