

太戈编程
etiger.vip

信奥算法

极差

请同学简述题意
突出核心要点

求一个整数数组的极差

算法

打擂台
求最大值

打擂台
求最小值



极差=最大值-最小值

代码1

```
8   cin >> n;
9   for (int i=0; i<n; i++)
10      cin >> a[i];
11   int big=a[0], small=a[0];
12   for(int i=1; i<n; i++){
13       if (a[i]>big)
14           big=a[i];
15       if (a[i]<small)
16           small=a[i];
17   }
18   cout << big - small <<endl;
```

代码2

```
1 #include<bits/stdc++.h>
2 using namespace std;
3 const int N=10009;
4 int n,a[N];
5 int main(){
6     freopen("range.in","r",stdin);
7     freopen("range.out","w",stdout);
8     cin >> n;
9     for(int i = 1;i <= n;i++) cin >> a[i];
10    int mx = *max_element(a + 1,a + n + 1);
11    int mn = *min_element(a + 1,a + n + 1);
12    cout << mx - mn << endl;
13    return 0;
14 }
```

1235

集训排名

请同学写出题目大意
已知什么求什么

有 n 场比赛,每场比赛有若干人的姓名和得分,请汇总成一份总分排名表,每行是一个人名+总分+每场得分

请同学阅读[数据规模和约定]
识别部分得分点

【数据规模与约定】

1,2号数据: $n=1$

3,4号数据: $n \leq 5$

5号数据: $x_i \leq 1$

所有数据: 保证 $n \leq 8000$, $x_i \leq 100$ 。总共参与的学员不超过100人。

模拟

根据题意逐步处理

难点：信息储存，输出格式

待明确：变量含义，数组含义

难点：信息储存

估算最终表格大小

行数=总人数 ≤ 100

列数=总比赛数 ≤ 10000

可以用二维数组储存每个人每场得分

`scores[i][j]`表示*i*号学生第*j*场的得分

参赛得了0分和没参赛的输出结果不同
如何区别?

表格里预先填充-1

`scores[i][j]`表示*i*号学生第*j*场的得分

难点



输入时学生只有姓名,并没有编号

人工分配编号

`nStu`表示当前发现了几位学生

```
map<string,int> id;  
每个姓名对应一个编号
```

```
struct Student{  
    string name;  
    int tot;  
} s[M];
```

```
71  memset(scores, -1, sizeof(scores));
72  int nStu=0;
73  for(int k=1; k<=nTests; k++){
74      int x;
75      cin>>x;
76      for(int i=1; i<=x; i++){
77          string name;
78          int score;
79          cin>>name>>score;
80          if(id.count(name))
81              
82          else{
83              id[name]=++nStu;
84              s[id[name]].name=name;
85              s[id[name]].tot=score;
86          }
87          
88      }
89  }
```

难点：输出格式

排版对齐

最长姓名有多长?

总分最高是几位数?

```
90  int mxLen=0;
91  for(int i=1;i<=nStu;i++)
92      mxLen=max(mxLen,(int)s[i].name.size());

93  sort(s+1,s+1+nStu,cmp);
94  int nDigits=count(s[1].tot);
```



```
95 for(int i=1;i<=nStu;i++){
96     printName(s[i].name,mxLen);
97     cout<<" ";
98     printScore(s[i].tot,nDigits);
99     cout<<":";
100     for(int k=1;k<=nTests;k++){
101         cout<<" ";
102         
103     }
104     cout<<endl;
105 }
```

太戈编程

2793



修电脑



请同学简述题意 突出核心要点

m 人里一个老板,共 n 台电脑要修理。
第 i 台需用时 $t[i]$ 。 $(m-1)$ 个员工工作时间相同，老板可以额外加班，
求最少多久全修完？

特殊条件的分析

10%数据, $m=1$

10%数据, $n=2, m=2$

10%数据, $n=3$

得到部分分

启发思路

10%数据, $m=1$

只有老板**1**人

答案为所有电脑修理时间总和

思考: 若 $m>n$ 会如何?

电脑不够每人分到**1**台
答案为所有电脑修理时间总和

10%数据， $n=2, m=2$

老板1人+员工1人

2台电脑修理时间 $t[1], t[2]$

每人负责1台电脑
答案为 $\max(t[1], t[2])$

二分枚举答案T

OK(T)判断能否在时间T以内完成所有修理

假设所有人共同修理时间x秒

$$(m - 1)x + T \geq \sum_{i=1}^n t[i]$$

$$\longleftrightarrow x \geq \left(\sum_{i=1}^n t[i] - T \right) / (m - 1)$$

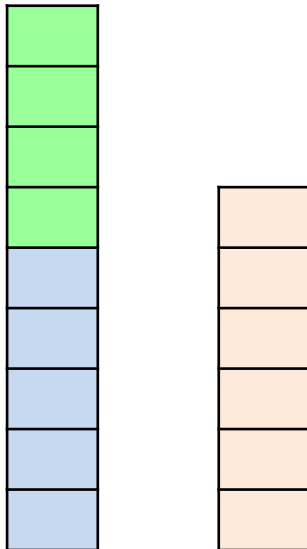
能否安排所有人共同修理至少x时间？

需要总耗时T越少，共同修理时间x就要越长

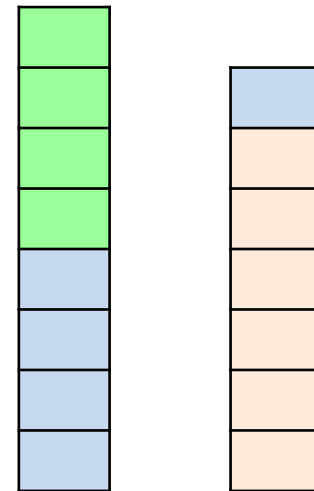
手算样例感受问题本质

共2人3台电脑, 每台的修理时间为4, 5, 6秒

能否安排所有人
共同修理时间6秒?



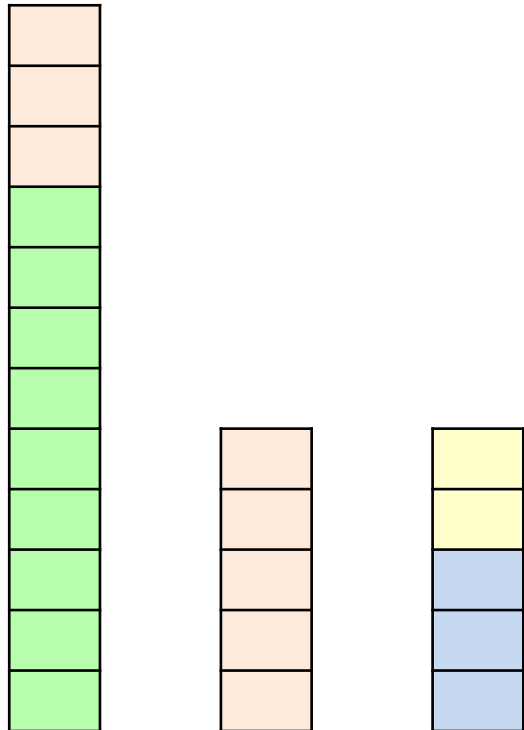
能否安排所有人
共同修理时间7秒?



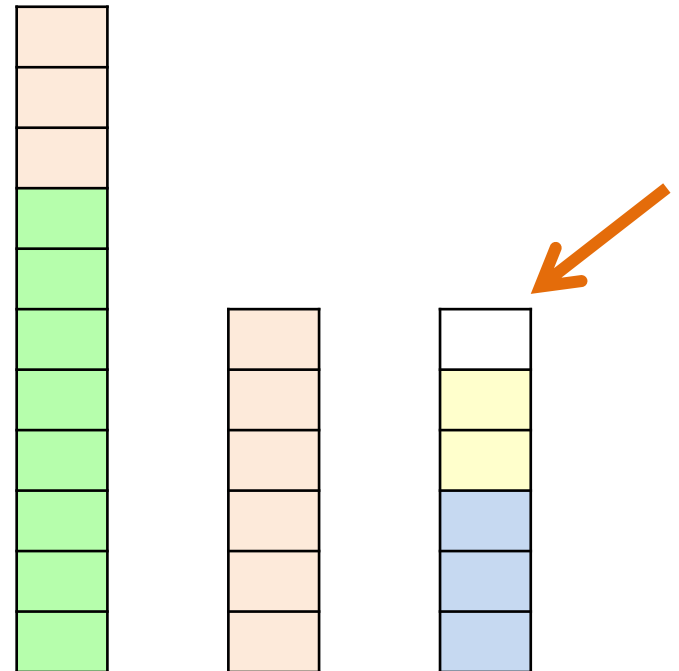
手算样例感受问题本质

共3人4台电脑, 每台的修理时间为2, 3, 8, 9秒

能否安排所有人
共同修理时间5秒?



能否安排所有人
共同修理时间6秒?



手算样例感受问题本质

共3人4台电脑, 每台的修理时间为2, 3, 8, 9秒

能否安排所有人
共同修理时间6?

不可能

3个人每人修时间6秒

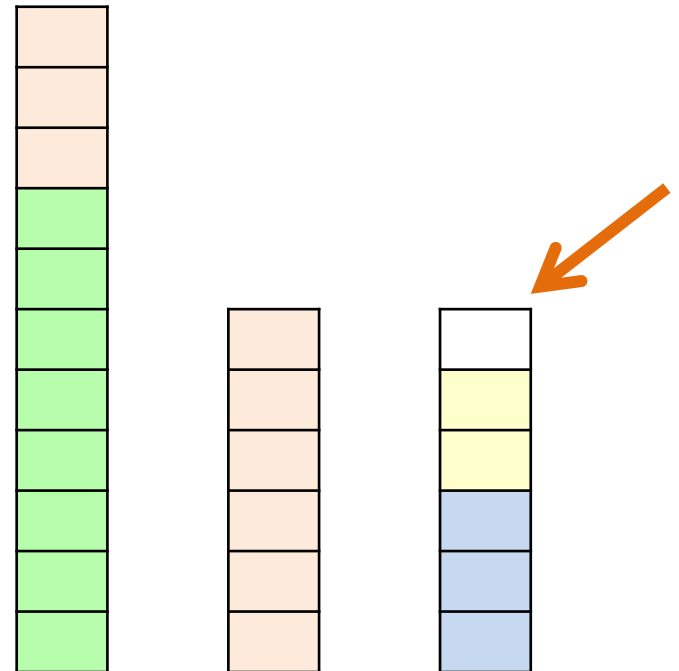
修8秒的电脑占满1个人的6秒

剩 $8-6=2$ 秒一定在6秒后发生

修9秒的电脑占满1个人的6秒

剩 $9-6=3$ 秒一定在6秒后发生

最后2秒3秒的电脑无法
占满1个人的6秒



如何判断能否安排所有人共同修理至少x时间？

给定x时，判断以下不等式是否成立

$$\sum_{i=1}^n \min(t[i], x) \geq mx$$

```
19 cin>>m>>n;
20 for(int i=1;i<=n;++i)cin>>t[i];
21 sort(t+1,t+1+n);
22 for(int i=1;i<=n;++i)s[i]=s[i-1]+t[i];

23 if(m>n||m==1){
24     cout<<fixed<<setprecision(3)<<s[n]<<endl;
25     return 0;
26 }

27 double l=0;
28 double r=s[n];
29 double ans=s[n];
30 while(r-l>ERR){
31     double mid=l+(r-l)/2;
32     if(OK(mid))ans=mid,r=mid-ERR;
33     else l=mid+ERR;
34 }
35 cout<<fixed<<setprecision(3)<<ans<<endl;
```

```
3 const int N=10009;  
4 const double ERR=0.0000001;  
5 double t[N],s[N];  
6 int n,m;
```

```
7 bool OK(double T){  
8     double x=(s[n]-T)/(m-1);  
9     if(x>T)return 0;  
10    double LHS=0;  
11    for(int i=1;i<=n;++i)  
12        LHS+=min(t[i],x);  
13    double RHS=m*x;  
14    return LHS>=RHS;  
15 }
```

左手边
left-hand side

右手边
right-hand side

$$\sum_{i=1}^n \min(t[i], x) \geq mx$$

$t[]$ 排序后
可以二分定位加速

```
7 bool OK(double T){  
8     double x=(s[n]-T)/(m-1);  
9     if(x>T) return 0;  
10    int k=lower_bound(t+1,t+1+n,x)-t;  
11    double LHS=s[k-1]+(n-k+1)*x;  
12    double RHS=m*x;  
13    return LHS>=RHS;  
14 }
```

 $x \leq t[k]$

$$\sum_{i=1}^n \min(t[i], x) \geq mx$$



2079



分葡萄

请同学写出题目大意
已知什么求什么

一棵树 n 个节点，要切断2条边，分成三块，
求最大块节点数-最小块节点数的差的最小值

请同学阅读[数据规模和约定]
识别部分得分点

【数据规模与约定】

1,2,3号数据: 保证 $3 \leq n \leq 200$ 。

4,5号数据: 保证 $3 \leq n \leq 2000$ 。

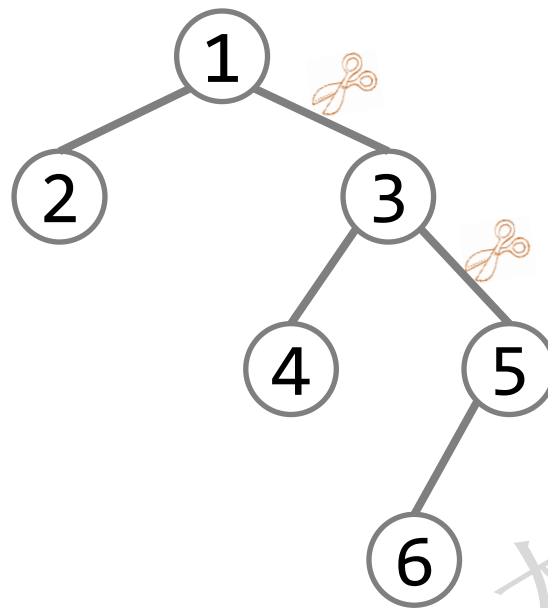
所有数据: 保证 $3 \leq n \leq 200000$ 。

输入

6
1 2
1 3
3 4
3 5
5 6

输出?

0



无根树 转 有根树

有根树上的边对应下方儿子节点

边信息可以下放到点信息

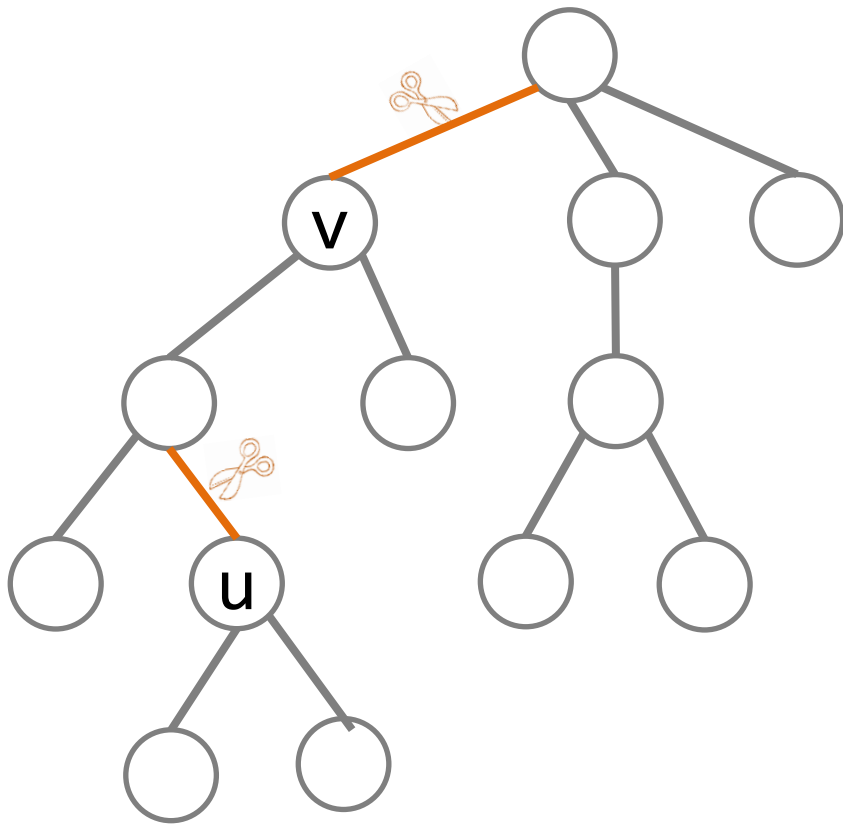
枚举删哪条边
可以通过枚举节点来进行

根节点不对应任何边
根节点不需要枚举

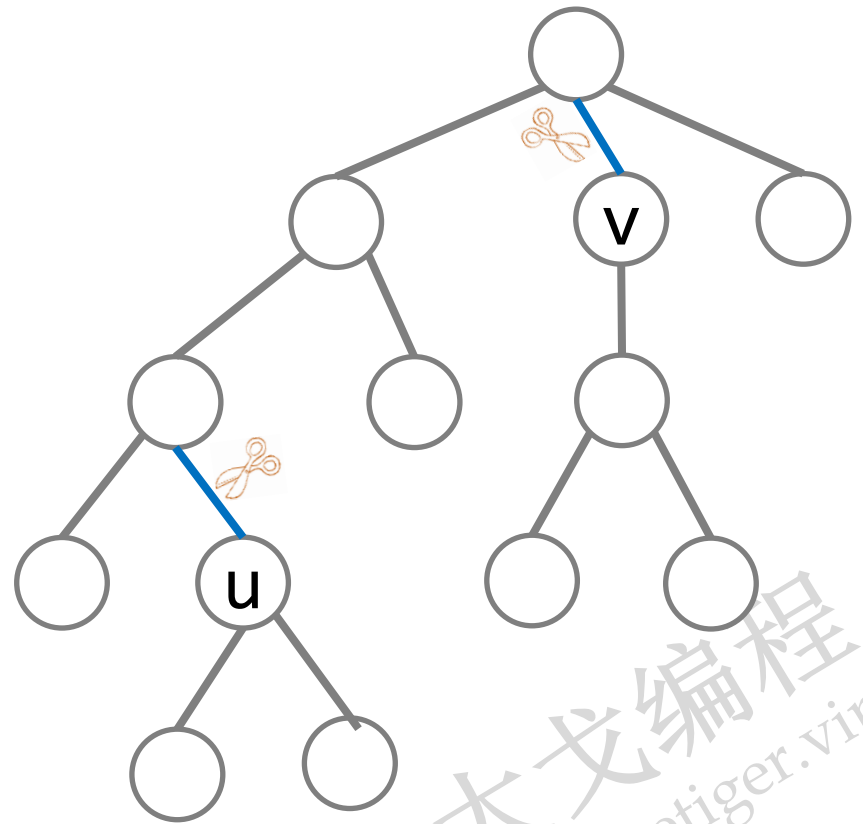
枚举两条边

$(u, p[u]), (v, p[v])$

分成的三块大小是多少?

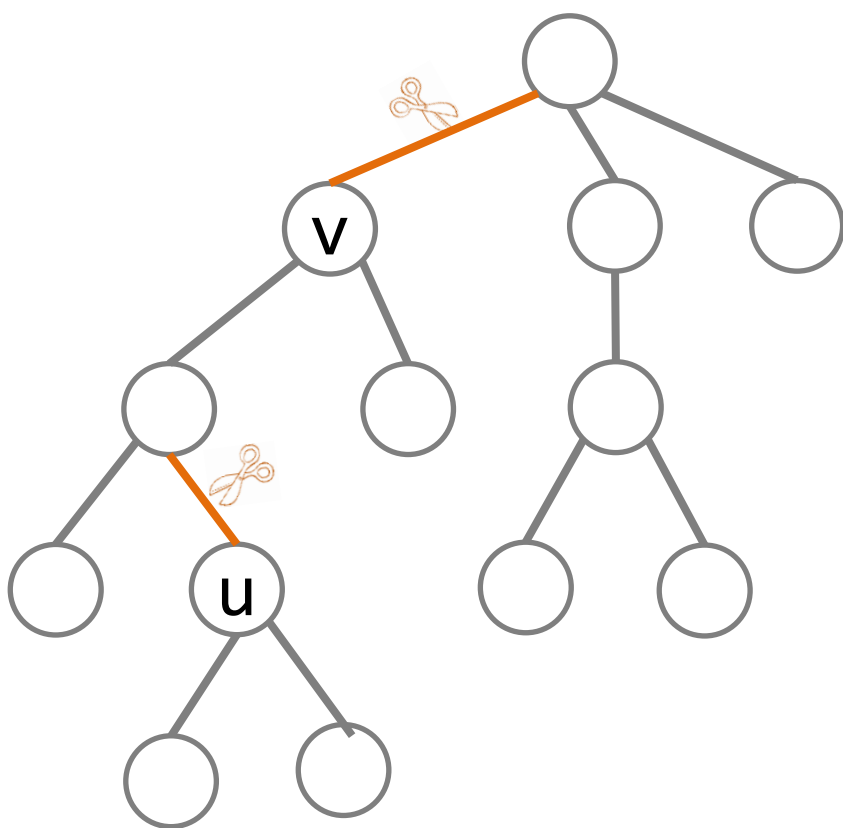


u, v 是祖孙关系



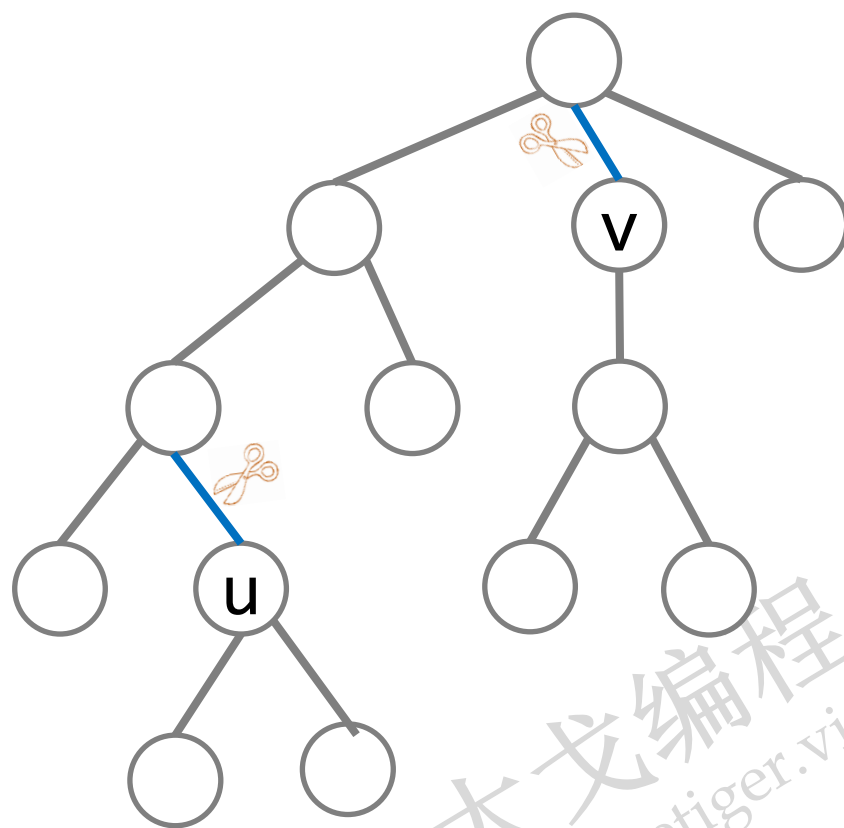
u, v 不是祖孙关系

$sz[u]$
 $sz[v] - sz[u]$
 $n - sz[v]$



u, v 是祖孙关系

$sz[u]$
 $sz[v]$
 $n - sz[v] - sz[u]$



u, v 不是祖孙关系

复杂度?

$O(n^2)$

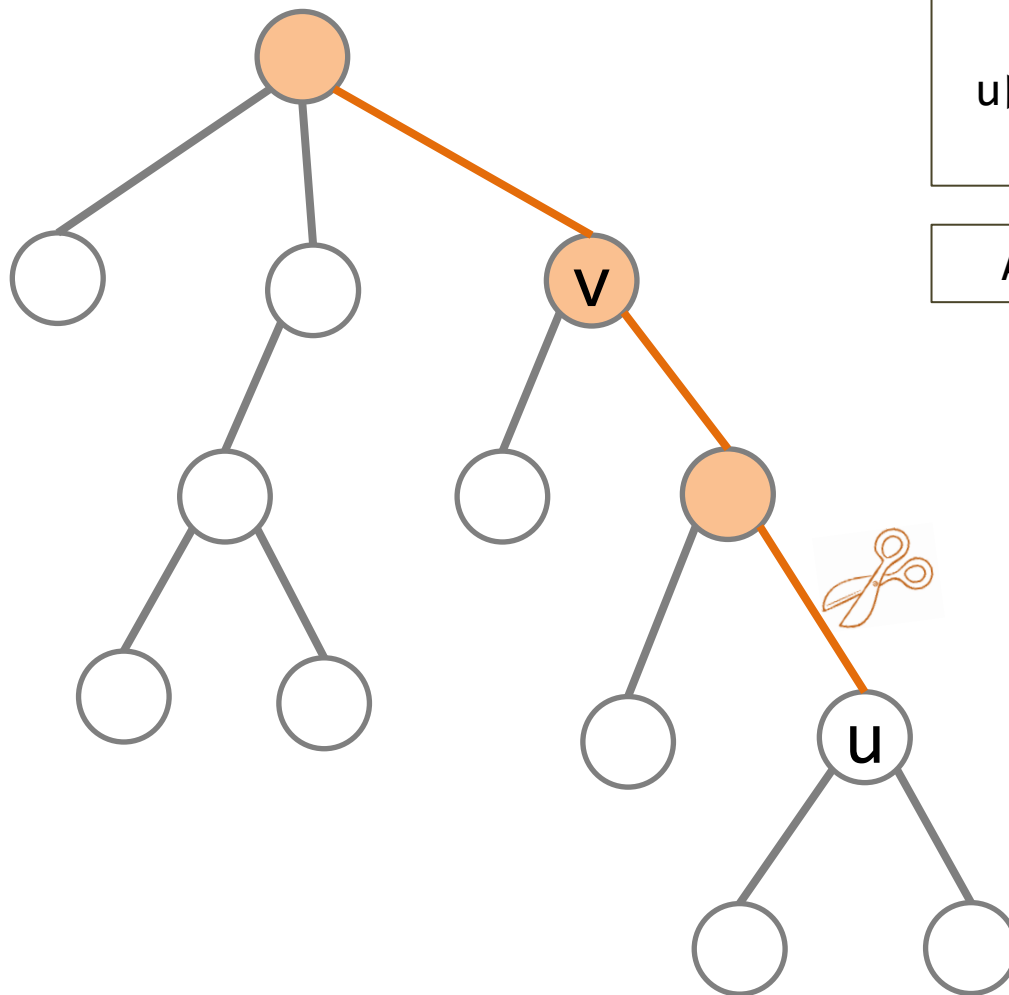
如何加速?

能否只枚举一条边
快速定位另一条边

u, v 是祖孙关系

枚举到u时
u的所有祖先sz值
放入A容器

A中做邻值查询


$$\begin{array}{l} \text{sz}[u] \text{ 固定} \\ \text{sz}[v] - \text{sz}[u] \\ n - \text{sz}[v] \end{array}$$

差值最小, 决策v

选 v 使 $n - sz[v]$
尽量接近
 $(n - sz[u]) / 2$

选 v 使 $sz[v]$
尽量接近
 $(n+sz[u])/2$

太戈编程
www.etiger.vip

u,v不是祖孙关系

枚举到u时
u的已访问的
非祖先sz值
放入B容器

B中做邻值查询

```
28 □ int diff(int a,int b,int c) {  
29   return max(max(a,b),c)-min(min(a,b),c);  
30 }
```

```
17 vector<int> closest(const set<int>& S,int v){
18     vector<int> result;
19     set<int>::iterator it=S.upper_bound(v);
20     if(it!=S.end()) result.push_back(*it);
21     if(it!=S.begin()) {
22         --it;
23         result.push_back(*it);
24     }
25     return result;
26 }
```

```

32 void solve(int u,int fa){
33     vector<int> v=closest(A,(n+sz[u])/2);
34     for(int i=0;i<v.size();i++){
35         int other_sz=v[i];
36         ans=min(ans,diff(sz[u],other_sz-sz[u],n-other_sz));
37     }
38     v=closest(B,(n-sz[u])/2);
39     for(int i=0;i<v.size();i++){
40         int other_sz=v[i];
41         
42     }
43     A.insert(sz[u]);
44     for(int i=0;i<to[u].size();i++) {
45         int v=to[u][i];
46         if(v==fa)continue;
47         solve(v,u);
48     }
49     A.erase(sz[u]);
50     B.insert(sz[u]);
51 }

```

太戈编程

2796