树的直径及其性质

树\$T= \langle E, V \rangle\$的直径定义为\$max\{ \delta(u,v) \} u,v \in V\$□其中\$\delta(u,v)\$表示点\$u\$和点\$v\$之间的简单路径。简单来说,在树上任取两个点可以得到它们之间的距离,而最大的那个距离就是直径。

树的直径有这些性质:

1.两端点一定是叶子节点。

证明:显然,如果有一个端点不是叶子则可以延长到一个叶子使直径边长。

2. 距任意点最远点一定是直径的端点。

证明:假设不是,我们从\$u\$找到的最远点是\$v\$\|\mathrm{n}\)面直径是从\$a\$到\$b\$\|\mathrm{如果\$v\$在\$\delta(a,b)\$上,那么显然距离\$u\$最远的点不是\$v\$\|\mathrm{v}\]还可以继续延长,矛盾。如果\$v\$不在\$\delta(a,b)\$上,那么我们在\$\delta(a,b)\$上选一个点\$p\$\|\mathrm{m}\]那么有\$dis(u,v)>dis(u,p)+dis(p,b)\$\|\mathrm{m}\]因此有\$dis(a,v) = dis(a,p) + dis(p,u) + dis(u,v) > dis(a,p) + dis(p,b) = dis(a,b)\$\|\mathrm{m}\]

3.两棵树相连,新直径的两端点一定是原四个端点中的两个,且新直径长度最小为max(max(直径1,直径2),半径1+半径2+新边长度)(设k为直径中最接近中点的节点,半径=max(tot-d[k],d[k]))[

证明:显然。

4.一棵树上接一个叶子结点,直径最多改变一个端点

证明:显然不可能改变两个端点,也有可能没有改变

树的直径求解

第一种方法是利用性质2,首先从任意一个点开始bfs□找到一个离这个点最远的点,这样就找到了直径的一端,然后我们从这个点再开始bfs就可以同时找出直径的长度和直径的两端。但这种方法的缺点是不能处理有负权的情况。代码实现十分简单,这里就不给出了。

第二种方法是树形DP \square 令 $\$f_1[i]$ \$表示节点\$i\$到它叶子节点路径长度的最大值 \square \$ $f_2[i]$ \$表示节点\$i\$到它叶子节点路径长度的次大值,我们只需要按照正常记录最大值和次大值的方法更新,最后的答案即为\$max $\{f_1[i] + f_2[i] \}$ \square 这一方法可以处理负权,但是很难确定直径的两端是哪两个节点。代码实现可以看这篇博客

树的直径例题

例题1——POJ1985

题目大意

树的直径裸题,不过题目输入非常规

解题思路

没有负权,两种方法皆可

代码实现

和上面的连接一样

例题2——Adjoin the Networks

传送门

题目大意

给出一个森林,求一种链接方式,将森林连成一棵树,并让树的直径最短。边权均为1。

解题思路

应用性质3,由于边权均为1,我们假设两棵树的直径长度分别为 \$a,b\$且 \$a \le b\$,则:

若\$ b \ge a+3 \$,则新树的直径长度仍为 \$ b \$,

若\$ b == a+2 \$, 当 \$b\$ 为奇数时,新树直径为 \$b+1\$, 当 \$b\$ 为偶数时,直径长度不改变。

若 \$ b == a+1 \$, 新树直径为 \$ b+1 \$□

经过以上分析,采取贪心的思想,我们只需要考虑前三大的直径就能保证直径不再扩大。其它的树相连后直径不会超过前三形成的直径。

代码实现

直接把林佬的代码偷过来

```
#include<algorithm>
#include<stack>
#include<ctime>
#include<cstring>
#include<cmath>
#include<iostream>
#include<iomanip>
#include<cstdio>
#include<queue>
using namespace std;
inline int read(){
```

https://wiki.cvbbacm.com/ Printed on 2025/11/04 15:09

```
int num=0,f=1;char x=getchar();
    while(x<'0'||x>'9'){if(x=='-')f=-1;x=getchar();}
    while (x>='0'\&\&x<='9') {num=num*10+x-'0'; x=getchar();}
    return num*f;
const int maxn=10005;
vector<int> e[maxn];
int n,m;
int vst[maxn],dis[maxn];
queue<int> Q;
int solve(int s){
    vst[s]=0;Q.push(s);
    int t=s,tmp=0;
    while(Q.size()){
        int x=Q.front();Q.pop();
        for(auto y:e[x]){
             if(vst[x]+1>vst[y]){
                 vst[y]=vst[x]+1;
                 Q.push(y);
                 if(vst[y]>tmp)tmp=vst[y],t=y;
            }
        }
    dis[t]=0;Q.push(t);tmp=0;
    while(Q.size()){
        int x=Q.front();Q.pop();
        for(auto y:e[x]){
             if(dis[x]+1>dis[y]){
                 dis[y]=dis[x]+1;
                 Q.push(y);
                 tmp=max(tmp,dis[y]);
        }
    printf("%d\n", tmp);
    return tmp;
int ans=0;
void get ans(int x){
    if(ans != 0)
        ans=\max(\text{ans}, (\text{ans}+1)/2+(x+1)/2+1);
    ans=max(ans,x);
int main(){
    n=read();m=read();
    for(int i=1,x,y;i<=m;++i){</pre>
        x=read()+1; y=read()+1;
        e[x].push_back(y);
        e[y].push back(x);
    for(int i=1;i<=n;++i)vst[i]=dis[i]=-1;</pre>
```

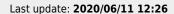
update: 2020-2021:teams:hotpot:diameterandweight https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:hotpot:diameterandweight&rev=1591849567 12:26

```
for(int i=1,t;i<=n;++i){</pre>
    if(vst[i] == -1)
         t=solve(i);
         get_ans(t);
printf("%d\n",ans);
return 0;
```

https://wiki.cvbbacm.com/ - CVBB ACM Team

Permanent link:

https://wiki.cvbbacm.com/doku.php?id=2020-2021: teams: hotpot: diameter and weight & rev=15918495672021 + teams: hotpot: diameter and weight & rev=1591849672021 + teams: hotpot: diameter and weight & rev=159184967200 + teams: hotpot: diameter and weight & rev=1591





https://wiki.cvbbacm.com/ Printed on 2025/11/04 15:09