

树的直径

树上任意两节点之间最长的简单路径即为树的「直径」。



前置知识：[树基础](#)。

引入

显然，一棵树可以有多条直径，他们的长度相等。

可以用两次 DFS 或者树形 DP 的方法在 $O(n)$ 时间求出树的直径。

例题

SPOJ PT07Z, Longest path in a tree



给定一棵 n 个节点的树，求其直径的长度。 $1 \leq n \leq 10^4$ 。

做法 1. 两次 DFS

过程

首先从任意节点 y 开始进行第一次 DFS，到达距离其最远的节点，记为 z ，然后再从 z 开始做第二次 DFS，到达距离 z 最远的节点，记为 z' ，则 $\delta(z, z')$ 即为树的直径。

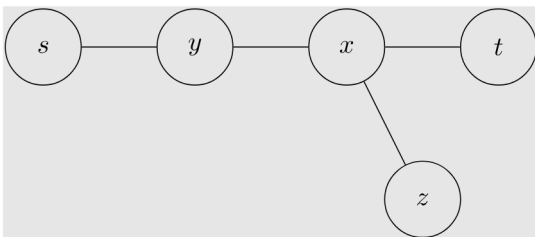
显然，如果第一次 DFS 到达的节点 z 是直径的一端，那么第二次 DFS 到达的节点 z' 一定是直径的一端。我们只需证明在任意情况下， z 必为直径的一端。

定理：在一棵树上，从任意节点 y 开始进行一次 DFS，到达的距离其最远的节点 z 必为直径的一端。

证明

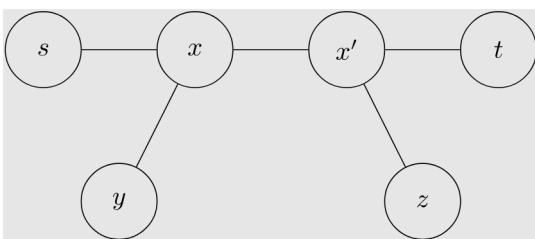
使用反证法。记出发节点为 y 。设真实的直径是 $\delta(s, t)$ ，而从 y 进行的第一次 DFS 到达的距离其最远的节点 z 不为 t 或 s 。共分三种情况：

- 若 y 在 $\delta(s, t)$ 上：



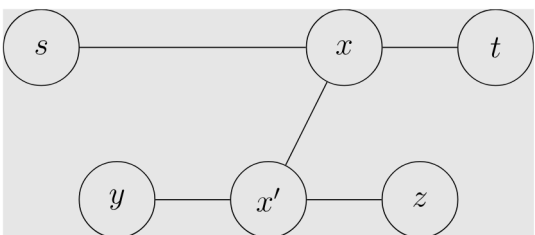
有 $\delta(y, z) > \delta(y, t) \implies \delta(x, z) > \delta(x, t) \implies \delta(s, z) > \delta(s, t)$ ，与 $\delta(s, t)$ 为树上任意两节点之间最长的简单路径矛盾。

- 若 y 不在 $\delta(s, t)$ 上，且 $\delta(y, z)$ 与 $\delta(s, t)$ 存在重合路径：



有 $\delta(y, z) > \delta(y, t) \implies \delta(x, z) > \delta(x, t) \implies \delta(s, z) > \delta(s, t)$ ，与 $\delta(s, t)$ 为树上任意两节点之间最长的简单路径矛盾。

- 若 y 不在 $\delta(s, t)$ 上，且 $\delta(y, z)$ 与 $\delta(s, t)$ 不存在重合路径：



有 $\delta(y, z) > \delta(y, t) \implies \delta(x', z) > \delta(x', t) \implies \delta(x, z) > \delta(x, t) \implies \delta(s, z) > \delta(s, t)$ ，与 $\delta(s, t)$ 为树上任意两节点之间最长的简单路径矛盾。

综上，三种情况下假设均会产生矛盾，故原定理得证。

⚠ 负权边

上述证明过程建立在所有路径均不为负的前提下。如果树上存在负权边，则上述证明不成立。故若存在负权边，则无法使用两次 DFS 的方式求解直径。

实现

代码实现如下。

```
1  constexpr int N = 10000 + 10;
2
3  int n, c, d[N];
4  vector<int> E[N];
5
6  void dfs(int u, int fa) {
7      for (int v : E[u]) {
8          if (v == fa) continue;
9          d[v] = d[u] + 1;
10         if (d[v] > d[c]) c = v;
11         dfs(v, u);
12     }
13 }
14
15 int main() {
16     scanf("%d", &n);
17     for (int i = 1; i < n; i++) {
18         int u, v;
19         scanf("%d %d", &u, &v);
20         E[u].push_back(v), E[v].push_back(u);
21     }
22     dfs(1, 0);
23     d[c] = 0, dfs(c, 0);
24     printf("%d\n", d[c]);
25     return 0;
26 }
```

如果需要求出一条直径上所有的节点，则可以在第二次 DFS 的过程中，记录每个点的前序节点，即可从直径的一端一路向前，遍历直径上所有的节点。

做法 2. 树形 DP

过程 1

我们记录当 1 为树的根时，每个节点作为子树的根向下，所能延伸的最长路径长度 d_1 与次长路径（与最长路径无公共边）长度 d_2 ，那么直径就是对于每一个点，该点 $d_1 + d_2$ 能取到的值中的最大值。

树形 DP 可以在存在负权边的情况下求解出树的直径。

实现 1

代码实现如下：

```
1  constexpr int N = 10000 + 10;
2
```

```

3  int n, d = 0;
4  int d1[N], d2[N];
5  vector<int> E[N];
6
7  void dfs(int u, int fa) {
8      d1[u] = d2[u] = 0;
9      for (int v : E[u]) {
10         if (v == fa) continue;
11         dfs(v, u);
12         int t = d1[v] + 1;
13         if (t > d1[u])
14             d2[u] = d1[u], d1[u] = t;
15         else if (t > d2[u])
16             d2[u] = t;
17     }
18     d = max(d, d1[u] + d2[u]);
19 }
20
21 int main() {
22     scanf("%d", &n);
23     for (int i = 1; i < n; i++) {
24         int u, v;
25         scanf("%d %d", &u, &v);
26         E[u].push_back(v), E[v].push_back(u);
27     }
28     dfs(1, 0);
29     printf("%d\n", d);
30     return 0;
31 }

```

如果需要求出一条直径上所有的节点，则可以在 DP 的过程中，记录下每个节点能向下延伸的最长路径与次长路径（定义同上）所对应的子节点，在求 d 的同时记下对应的节点 u ，使得 $d = d_1[u] + d_2[u]$ ，即可分别沿着从 u 开始的最长路径的次长路径对应的子节点一路向某个方向（对于无根树，虽然这里指定了 1 为树的根，但仍需记录每点跳转的方向；对于有根树，一路向上跳即可），遍历直径上所有的节点。

过程 2

这里提供一种只使用一个数组进行的树形 DP 方法。

我们定义 $dp[u]$ ：以 u 为根的子树中，从 u 出发的最长路径。那么容易得出转移方程：

$dp[u] = \max(dp[u], dp[v] + w(u, v))$ ，其中的 v 为 u 的子节点， $w(u, v)$ 表示所经过边的权重。

对于树的直径，实际上是通过枚举从某个节点出发不同的两条路径相加的最大值求出。因此，在 DP 求解的过程中，我们只需要在更新 $dp[u]$ 之前，计算 $d = \max(d, dp[u] + dp[v] + w(u, v))$ 即可算出直径 d 。

实现 2

代码实现如下：

```

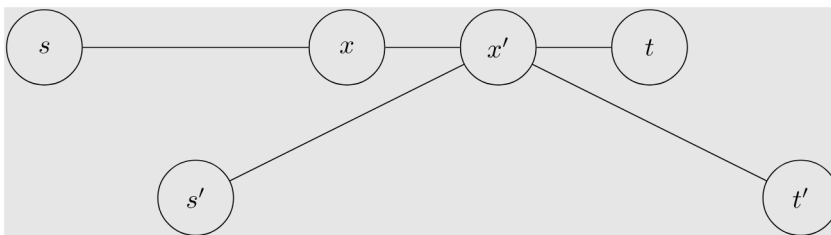
1  constexpr int N = 10000 + 10;
2
3  int n, d = 0;
4  int dp[N];
5  vector<int> E[N];
6
7  void dfs(int u, int fa) {
8      for (int v : E[u]) {
9          if (v == fa) continue;
10         dfs(v, u);
11         d = max(d, dp[u] + dp[v] + 1);
12         dp[u] = max(dp[u], dp[v] + 1);
13     }
14 }
15
16 int main() {
17     scanf("%d", &n);
18     for (int i = 1; i < n; i++) {
19         int u, v;
20         scanf("%d %d", &u, &v);
21         E[u].push_back(v), E[v].push_back(u);
22     }
23     dfs(1, 0);
24     printf("%d\n", d);
25     return 0;
26 }

```

性质

若树上所有边边权均为正，则树的所有直径中点重合

证明：使用反证法。设两条中点不重合的直径分别为 $\delta(s, t)$ 与 $\delta(s', t')$ ，中点分别为 x 与 x' 。显然， $\delta(s, x) = \delta(x, t) = \delta(s', x') = \delta(x', t')$ 。




有 $\delta(s, t') = \delta(s, x) + \delta(x, x') + \delta(x', t') > \delta(s, x) + \delta(x, t) = \delta(s, t)$ ，与 $\delta(s, t)$ 为树上任意两节点之间最长的简单路径矛盾，故性质得证。

习题

- [CodeChef, Diameter of Tree](#)
- [Educational Codeforces Round 35, Problem F, Tree Destruction](#)

- [ZOJ 3820 Building Fire Stations](#)
- [CEOI2019/CodeForces 1192B. Dynamic Diameter](#)
- [ICPC 2019 上海赛区网络赛 Lightning Routing I](#)
- [NOIP2007 提高组 树网的核](#)
- [SDOI2011 消防](#)
- [APIO2010 巡逻](#)

 本页面最近更新：2025/8/29 18:05:34，[更新历史](#)

 发现错误？想一起完善？ [在 GitHub 上编辑此页！](#)

 本页面贡献者：[StudyingFather](#), [lychees](#), [Mout-sea](#), [Tiphereth-A](#), [countercurrent-time](#), [Enter-tainer](#), [H-J-Granger](#), [Haohu Shen](#), [NachtgeistW](#), [sshwy](#), [383494](#), [AngelKitty](#), [CCXXI](#), [cjsoft](#), [diauweb](#), [Early0v0](#), [ezoixx130](#), [GavinZhengOI](#), [GekkaSaori](#), [Gesruea](#), [HeRaNO](#), [iamtwz](#), [Ir1d](#), [Kaiser-Yang](#), [Konano](#), [kxccc](#), [LovelyBuggies](#), [Makkiy](#), [mcendu](#), [mgt](#), [minghu6](#), [ouuan](#), [P-Y-Y](#), [Peanut-Tang](#), [PeterlitsZo](#), [PotassiumWings](#), [SamZhangQingChuan](#), [sbofgayschool](#), [SukkaW](#), [Suyun514](#), [weiyong1024](#), [zcz0263](#)

© 本页面的全部内容 [在 CC BY-SA 4.0 和 SATA 协议之条款下](#) 提供，附加条款亦可能应用