提高算法班

树形DP

Mas

#2861、没有上司的舞会

题目描述

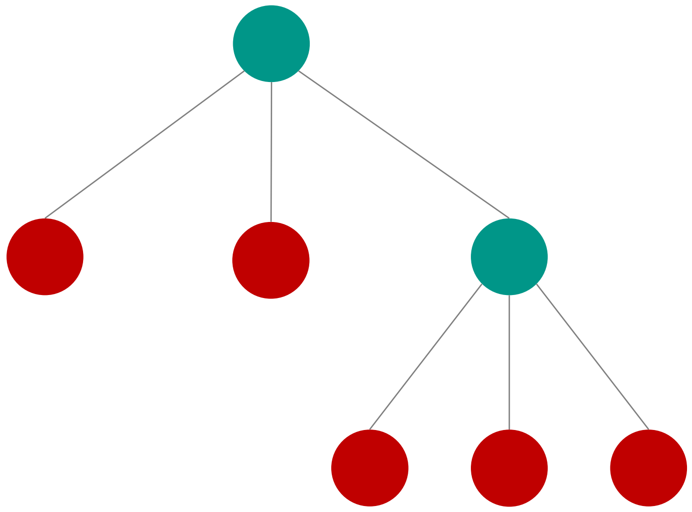
Ural 大学有 N 个职员,编号为 $1 \sim N$.

他们有从属关系,也就是说他们的关系就像一棵以校长为根的树
父结点就是子结点的直接上司

每个职员有一个快乐指数

现在有个周年庆宴会,要求与会职员的快乐指数最大

> *但是,没有职员愿和直接上司一起与会*



输入格式

第一行一个整数 N

接下来 N 行
第 $i+1$ 行表示 i 号职员的快乐指数 R_i

接下来 $N-1$ 行,每行输入一对整数 L, K
表示 K 是 L 的直接上司

最后一行输入 `0 0`

输出格式

输出最大的快乐指数

数据规模

对于全部的数据 $1 \leq N \leq 6000, -128 \leq R_i \leq 127$

#2861、没有上司的舞会

设 $\mathrm{dp}[u][0]$ 为 u 为子树根时且 u 不取时的最大快乐值

设 $\mathrm{dp}[u][1]$ 为 u 为子树根时且 u 取时的最大快乐值

$\mathrm{dp}[u][0/1]$ 与 $\mathrm{dp}[v][0/1]$ 有关其中 $v \in \mathrm{son}_u$

若 u 不取那么其子节点可取可不取

$$\mathrm{dp}[u][0] = \sum_{v \in \mathrm{son}_u} \max(\mathrm{dp}[v][1], \quad \mathrm{dp}[v][0])$$

如果节点 u 取，那么其子节点不能取

$$\mathrm{dp}[u][1] = w_u + \sum_{v \in \mathrm{son}_u} dp[v][0]$$

答案为 $\max(\mathrm{dp}[\mathrm{root}][0], \mathrm{dp}[\mathrm{root}][1])$，时间复杂度 $O(n)$

#741、战略游戏

题目描述

Bob 喜欢玩电脑游戏,特别是战略游戏

但是他经常无法找到快速玩过游戏的方法, 现在他有个问题:
现在他有座古城堡,古城堡的路形成一棵树。他要在这棵树的节点上放置最少数目的士兵,使得这些士兵能够瞭望到所有的路

> *某个士兵在一个节点上时,与该节点相连的所有边都将能被瞭望到*

请你编一个程序,给定一棵树,帮 Bob 计算出他最少要放置的士兵数

输入格式

输入数据表示一棵树,描述如下

第一行一个数 N ,表示树中节点的数目

第二到第 $N+1$ 行,每行描述每个节点信息,依次为该节点编号 i ,数值 k , k 表示后面有 k 条边与节点 i 相连,接下来 k 个数,分别是每条边的所连节点编号 $r_1, r_2, \cdots, r_k$

对于一个有 N 个节点的树,节点标号在 0 到 $N-1$ 之间,且在输入中每条边仅出现一次

输出格式

输出仅包含一个数,为所求的最少士兵数

数据范围

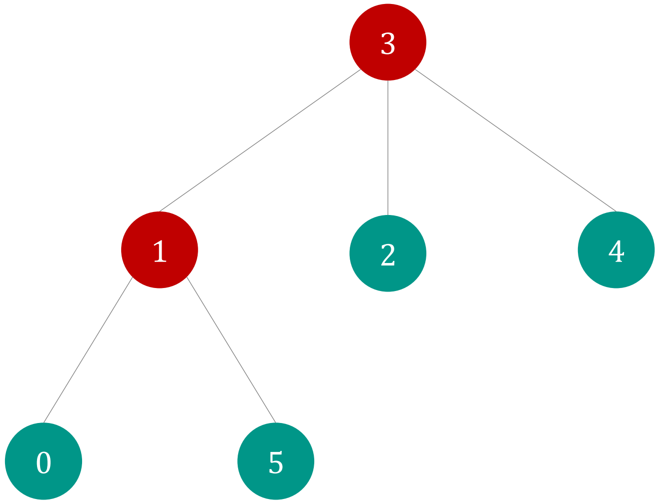
对于 100% 的数据 $0 < N \leq 1500$

#741、战略游戏

设 dp[u][0] 为考虑以 u 为根的子树 且 u 不选时所需最少点数

设 dp[u][1] 为考虑以 u为根的子树 且 u 选时所需最少点数

当 u 不选时，其出边必须被覆盖



$$\mathrm{dp}[u][0] = \sum_{v \in \mathrm{son}_u} \mathrm{dp}[v][1]$$

当 u 选时，其出边可被子节点覆盖也可不被覆盖

$$\mathrm{dp}[u][1] = 1 + \sum_{v \in son_u} \min(\mathrm{dp}[v][0], \mathrm{dp}[v][1])$$

答案为 min(dp[root][0], dp[root][1])

时间复杂度 $O(n)$

#742、皇宫看守

题目描述

太平王世子事件后,陆小凤成了皇上特聘的御前一品侍卫

皇宫以午门为起点,直到后宫嫔妃们的寝宫,呈一棵树的形状,某些宫殿间可以互相望见
大内保卫森严,三步一岗,五步一哨,每个宫殿都要有人全天候看守,在不同的宫殿安排看守所需的费用不同

可是陆小凤手上的经费不足,无论如何也没法在每个宫殿都安置留守侍卫

帮助陆小凤布置侍卫,在看守全部宫殿的前提下,使得花费的经费最少

输入格式

第一行 n ,表示结点的数目

第二 $2 \sim n+1$ 行,每行描述每个宫殿结点信息
依次为：该宫殿结点标号 i ,在该宫殿安置侍卫所需的经费 k ,该边的儿子数 m

接下来 m 个数,分别是这个节点的 m 个儿子的标号 $r_1, r_2, \cdots, r_m$

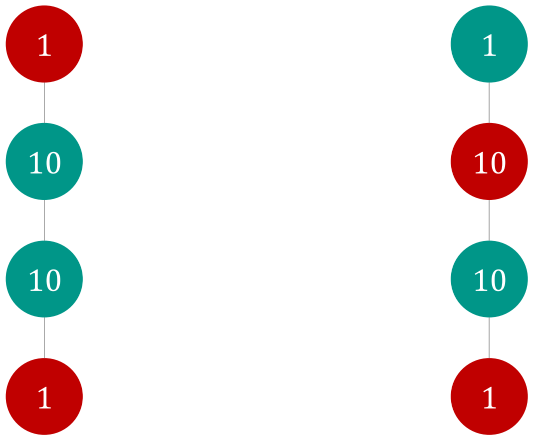
对于一个 n 个结点的树,结点标号在 1 到 n 之间,且标号不重复

输出格式

输出最少的经费

数据范围

对于 100% 的数据, $0 < n \le 1500$



左图为符合本题要求的选取方式，最小代价 $1+1=2$

右图为符合上题要求的选取方式，最小代价 $10+1=11$

上题关注边是否覆盖，而本题关注点的选取

权值在点上，本题也非带权最小边覆盖

若状态设计为 $\text{dp}[u][0/1]$ 也无法转移

父节点 u 的状态受限于子节点 v，也受限于 u 的父节点

#742、皇宫看守

设 $\mathrm{dp}[u][0]$ 为将 u 为根的子树覆盖，u 由父节点覆盖的最小代价和

设 $\mathrm{dp}[u][1]$ 为将 u 为根的子树覆盖，u 由自己覆盖的最小代价和

设 $\mathrm{dp}[u][2]$ 为将 u 为根的子树覆盖，u 由子节点覆盖的最小代价和

记 $w[u]$ 为 u 放置守位的代价

- 节点 u 不设置守卫由父节点覆盖，其子节点 v 只能由自己或 v 的子节点覆盖

$$\mathrm{dp}[u][0] = \sum_{v \in \mathrm{son}_u} \min(\mathrm{dp}[v][1], \mathrm{dp}[v][2])$$

- 节点 u 设置守卫，其子节点 v 三种情况皆可

$$\mathrm{dp}[u][1] = w[u] + \sum_{v \in \mathrm{son}_u} \min(\mathrm{dp}[v][0], \mathrm{dp}[v][1], \mathrm{dp}[v][2])$$

#742、皇宫看守

- 节点 u 不设置守卫由子节点覆盖

  $\text{dp}[u][0]$ 中包含 u 未付出代价且 u 的子树被覆盖的最小代价

  不妨枚举覆盖的子节点 k

  若被子节点 k 覆盖，减去 k 为根的子树覆盖的代价

$$\min(\text{dp}[k][1]\ ,\text{dp}[k][2])$$

  再强制加上 k 自己放置守卫并覆盖其子树的代价 $\text{dp}[k][1]$

$$\text{dp}[u][2] = \min_{k \in \text{son}_u} \{\, \text{dp}[u][0] - \min(\text{dp}[k][1], \text{dp}[k][2]) + \text{dp}[k][1] \,\}$$

  在计算出 $\text{dp}[u][0]$ 后再计算 $\text{dp}[u][2]$

对于叶节点 v 有 $\text{dp}[v][0] = 0$, $\text{dp}[v][1] = w[v]$, v 并不存在子节点所以 $\text{dp}[v][2] = \infty$

显然根节点没有父亲，答案为 $\min(\text{dp}[\text{root}][1]\ ,\text{dp}[\text{root}][2])$

时间复杂度 $O(n)$

独立集 & 点覆盖 & 支配集

独立集

对于图 G = (V,E)，若 V′ ∈ V 且 V′ 中任意两点都不相连

则称 V' 是图 G 的一个**独立集**（independent set）

二分图的最大独立集 = 点数 - 二分图的最大匹配

#2861、没有上司的舞会 为树上带权最大独立集

将树根据 DFS 层次 奇/偶 将点划分成两个部分，那么可将树看作二分图

若为不带权最大独立集，可使用 Hungarian Algorithm 解决

时间复杂度 $O(|V|\,|E|)$

点覆盖

对于图 G = (V,E)，若 V' ∈ V 且 $\forall e$ ∈ E 满足 e 至少一个端点在 V' 中

则称 V' 是 G 的一个 **点覆盖** (vertex cover)

独立集 & 点覆盖 & 支配集

二分图的最小点覆盖 = 二分图的最大匹配

#741、▯ 略游▯ 为树上最小点覆盖， Hungarian Algorithm 求出最大匹配即为答案

时间复杂度 $O(|V|\,|E|)$

支配集

对于图 $G = (V, E)$，若 $V' \in V$ 且 $\forall v \in \{V \setminus V'\}$ 存在边 $(u, v) \in E$ 满足 $u \in V'$

那么 V' 是 G 的一个 **支配集** (dominating set)

#742、皇▯ 看守 为树上带权最小支配集

对于树上不带权最小支配集可贪心求解

#1819、又是树直径

题目描述

给一个 n 节点的边权树,现在请你找到树中的一条最长路径

换句话说,要找到一条路径,使得使得路径两端的点的距离最远

> *路径中可以只包含一个点*

输入格式

第一行输入一个正整数 n

接下来 $n-1$ 行,每行三个数: u, v, w

表示 $u \to v$ 有一条权重 w 的边

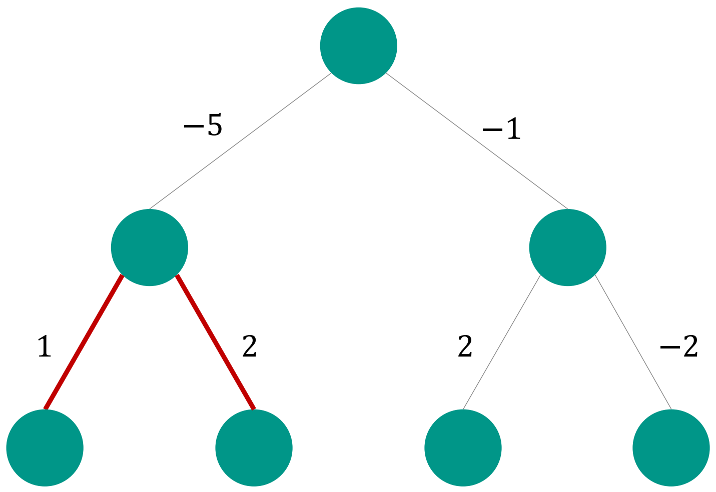
输出格式

输出一个整数,最长路径

数据规模

对于 40% 的数据 $1 \le n \le 100$

对于 100% 的数据 $1 \le n \le 200000, -1000 \le w_i \le 1000$



树的直径有两种常用求法

- 两次 DFS/BFS
- 树形DP

其中两次 DFS/BFS 仅适用于**非负边权**树

两次 DFS/BFS 实现:

从 **任意** 一点 P 出发，通过 DFS/BFS 寻找离它最远的点 Q

树的直径

- 再次从点 Q 出发，通过 DFS/BFS 寻找离它最远的 W

直径即为 WQ 距离，时间复杂度 $O(|V| + |E|)$

边权非负两次 DFS/BFS 正确性证明

若点 P 在直径上

根据的定义 Q 必为直径的一个端点

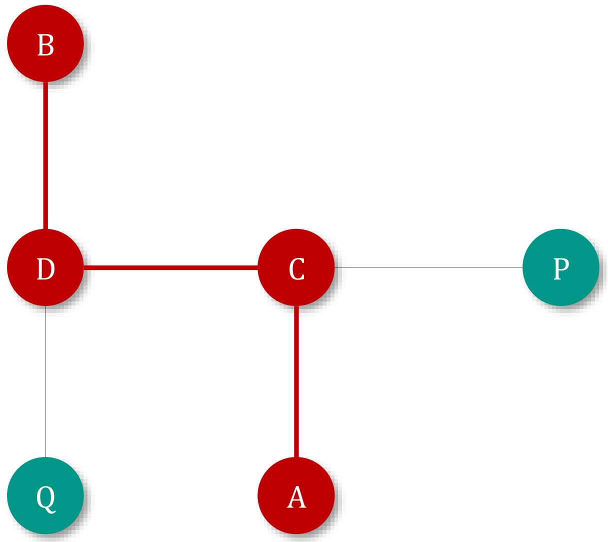
若非如此 PQ + PW 能组成一条更长的简单路径，与定义矛盾

若点 P 不在直径上

设直径为 AB，若 Q 不在 AB 上，有以下两种情况：

- AB 与 PQ 有交点 C

根据上述 DFS/BFS 规则有



$$PC + CD + DQ > PC + CD + DB \Rightarrow CD + DQ > CD + DB$$

$$\Rightarrow AC + CD + DQ > AC + CD + DB$$

树的直径

即 AQ > AB，不符合直径定义，矛盾

- AB 与 PQ 无交点

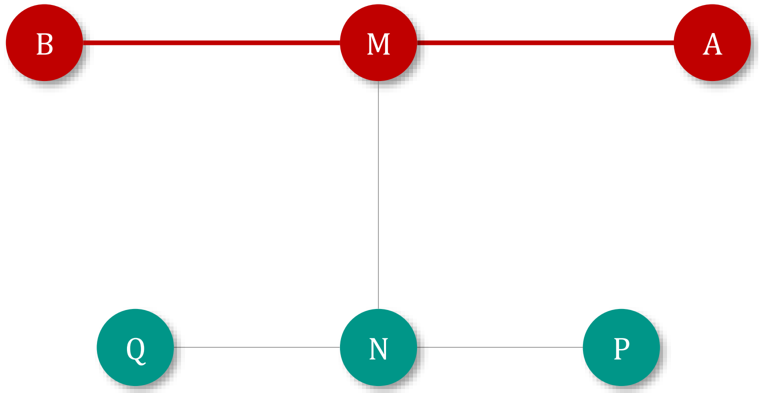
M 为 AB 上任意一点，N 为 PQ 上任意一点

根据上述 DFS/BFS 规则有

$$\mathrm{PN} + \mathrm{NQ} > \mathrm{PN} + \mathrm{NM} + \mathrm{MB} \Longrightarrow \mathrm{NQ} > \mathrm{NM} + \mathrm{MB}$$

$$\Longrightarrow \mathrm{NQ} + \mathrm{MN} + \mathrm{MA} > \mathrm{NM} + \mathrm{MB} + \mathrm{MN} + \mathrm{AM}$$

即 PQ > AB + 2NM，不符合直径定义，矛盾



当树上距离存在负边权，该结论并不成立，即无法 两次 DFS/BFS 求出树的直径

考虑求出从 u 出发终点位于 u 子树内的最大路径和，设其为 dp_u

$$\mathrm{dp}_u = \max_{v \in \mathrm{son}_u} (\mathrm{dp}_v + w_{u,v})$$

#1819、又是树直径

对于经过 u 的最长路径存在以下情况

- 未横跨节点 u

  此时路径一端为 u 另一端在 u 的子树内，最长路径和即 dp_u

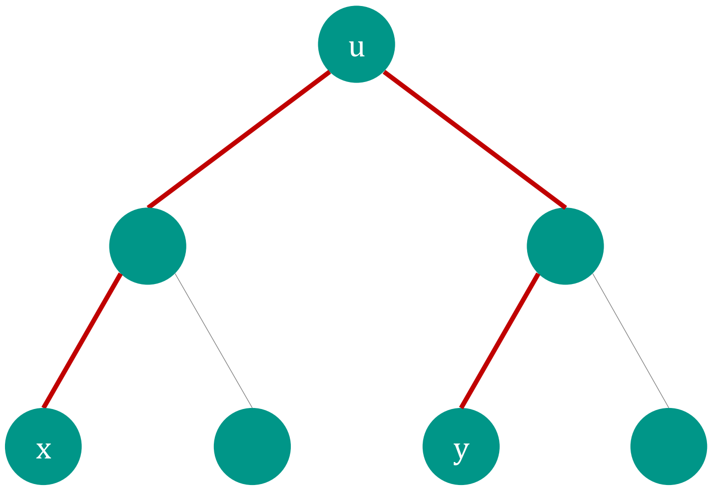
- 横跨节点 u

  若终点为 x,y 即 $\mathrm{LCA}(x,y)=u$

  子树的最大路径和加上边权 d_1，仅需令 $d_1 = \max\limits_{v\in \mathrm{son_u}}(\mathrm{dp}_v + w_{u,v})$

  若 d_1 对应节点为 x，不经过 v 的子树的次大路径加上边权 d_2，仅需令 $\mathrm{d}_2 = \max\limits_{v\in \mathrm{son_u} \wedge v\neq x}(\mathrm{dp}_v + w_{u,v})$

  两者之和即为横跨节点 u 的最长路径



对所有节点的两种情况取最大值记为 D

max(D , 0) 即为答案，时间复杂度 $O(n)$

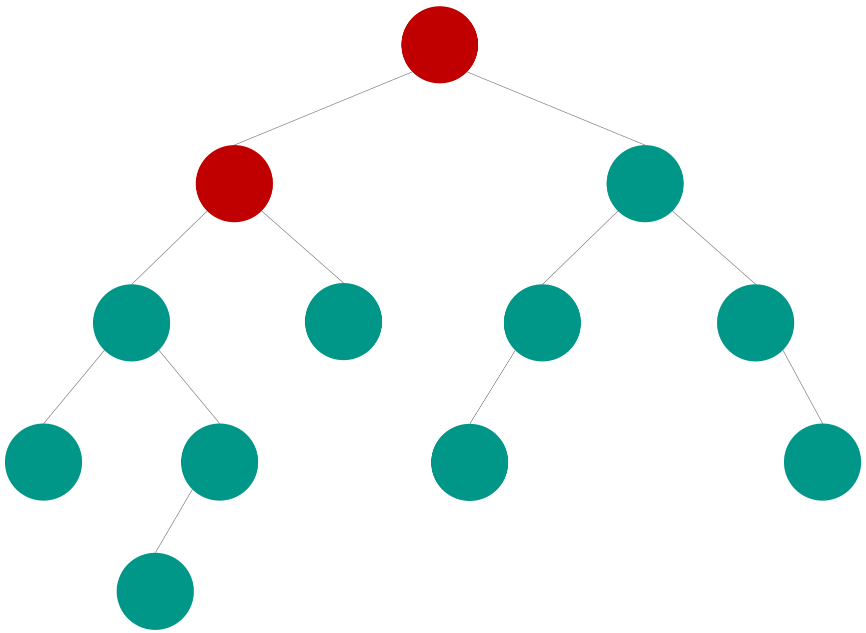
#2658、树的重心

题目描述

树中包含 n 个结点（编号 $1 \sim n$ ）和 $n-1$ 条无向边

请你找到树的重心

重心是指树中的一个结点,如果将这个点删除后
剩余各个连通块中点数的**最大值最小**,那么这个节点被称为树的重心



输入格式

第一行包含整数 n ,表示树的结点数

接下来 $n-1$ 行,每行包含两个整数 a 和 b
表示点 a 和点 b 之间存在一条边

输出格式

输出树的重心编号,如果重心不唯一从小到大输出

数据范围

对于全部的数据 $1 \le n \le 10^5$

#2658、树的重心

令 dp_u 表示删除 u 后**最大**联通块的大小

cnt_u 表示以 u 为根的子树的节点数量

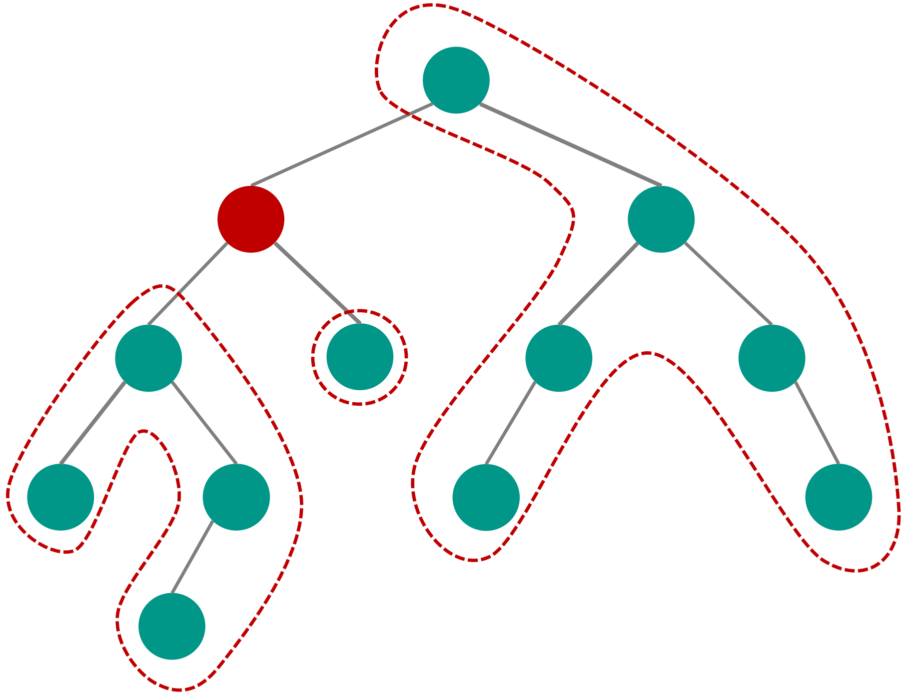
$$\mathrm{cnt}_u = 1 + \sum_{v \in \mathrm{son}_u} \mathrm{cnt}_v$$

除以 u 为根的子树的节点数量为 $n - cnt_u$

$$\mathrm{dp}_u = \max\left\{\max_{v \in \mathrm{son}_u}\{\mathrm{cnt}_v\}, n - \mathrm{cnt}_u\right\}$$

当 dp_u 取得最小值时，u 为树重心

时间复杂度 $O(n)$



树的重心

树的重心性质

- 树的重心若不唯一，则至多有两个且这两个重心相邻
- 以树的重心为根时，所有子树的大小都不超过整棵树大小的一半
- 树中所有点到某个点的距离和中，到重心的距离和最小

  若有两个重心，那么到它们的距离和一样
- 把两棵树通过一条边相连得到一棵新的树，那么新的树的重心在连接原来两棵树的重心的路径上
- 在一棵树上添加或删除一个叶子，那么它的重心最多只移动一条边的距离

证明见 树的重心的性质及其证明

#739、选课

题目描述

大学实行学分制,每门课程都有一定的学分,学生只要选修了这门课并通过考核就能获得相应学分

学生最后的学分是他选修各门课的学分总和，每个学生都要选择规定数量的课程

有些课程可以直接选修,有些课程需要一定的基础知识,必须在选了其他的一些课程基础上才能选修

例如《数据结构》必须在选修了《高级语言程序设计》后才能选修

我们称《高级语言程序设计》是《数据结构》的先修课,每门课的直接先修课最多只有一门,两门课也可能存在相同的先修课

为便于表述,每门课都有一个课号,课号依次为 $1,2,3,\cdots$

下面举例说明:

课号	先修课号	学分
1	无	1
2	1	1
3	2	3
4	无	3
5	2	4

上例中课号 1 是课号 2 的先修课,即如果要先修课号 2 ,则课号 1 必定已被选过
同样,如果要选修课号 3 ,那么课号 1 和 课号 2 都一定被选修过

学生不可能学完大学开设的所有课程,因此必须在入学时选定自己要学的课程

每个学生可选课程的总数是给定的
请找出一种选课方案使得你能得到的学分最多,并满足先修课优先的原则
假定课程间不存在时间上的冲突

数据范围

对于全部的数据 $1 \le N \le M \le 100$,学分不超过 20

输入格式

第一行两个正整数 M,N ,分别表示待选课程数和可选课程数

接下来 M 行每行描述一门课,课号依次为 $1,2,\cdots,M$

每行两个数,依次表示这门课先修课课号(若不存在,则该项值为 0)和该门课的学分

输出格式

输出一行,表示实际所选课程学分之和

#739、选课

```cpp
void dfs(int u)
{
  dp[u][1] = v[u];
  for (int i = head[u]; i; i = e[i].next)
  {
    int v = e[i].v;
    dfs(v);
    for (int j = m; j >= 0; j--)
      for (int k = 1; k <= j - 1; k++)
        dp[u][j] = max(dp[u][j], dp[u][j - k] + dp[v][k]);
  }
}
```

课程间依赖构成森林，增加超级点 0 转换成 $\mathrm{N}+1$ 个节点的树

设 $\mathrm{dp}[u][i][j]$ 为以 u 为根的子树考虑前 i 棵子树选了 j 门课的最高学分

初始时 $\mathrm{dp}[u][0][1] \longleftarrow w[u]$ 即选择课程 u

对于 u 的子节点 v，可将 【以 v 为根的子树所有选取方案】视作一个**分组**

组内有若干重量不同的物品 且 组内仅能选择一个物品

同时应保证选取重量和小于 j（需给课程 u 保留空间）

$$\mathrm{dp}[u][i][j] = \max\left(\begin{array}{c} \mathrm{dp}[u][i-1][j] \\ \max\limits_{v \in \mathrm{son}_u \wedge 0 \le k < j} \{ \mathrm{dp}[u][i-1][j-k] + \mathrm{dp}[v][\,|\mathrm{son}_v|\,][k] \} \end{array} \right)$$

答案为 $\mathrm{dp}[0][|\mathrm{son}_0|][\mathrm{N}+1]$,第二维度可优化

每次合并时间复杂度 $O(\mathrm{M}^2)$，总时间复杂度 $O(\mathrm{NM}^2)$

时间复杂度可优化至 $O(\mathrm{NM})$

#1287、金明的预算方案

实验舱 青少年编程 LAB 走近科学 走进名校

题目描述

金明今天很开心,家里购置的新房就要领钥匙了,新房里有一间金明自己专用的很宽敞的房间

更让他高兴的是,妈妈昨天对他说："你的房间需要购买哪些物品,怎么布置,你说了算,只要不超过 N 元钱就行"

今天一早,金明就开始做预算了,他把想买的物品分为两类：主件与附件,附件是从属于某个主件的,下表就是一些主件与附件的例子：

主件	附件
电脑	打印机,扫描仪
书柜	图书
书桌	台灯,文具
工作椅	无

如果要买归类为附件的物品,必须先买该附件所属的主件

每个主件可以有 0 个、 1 个或 2 个附件
附件不再有从属于自己的附件

金明想买的东西很多,肯定会超过妈妈限定的 N 元

于是,他把每件物品规定了一个重要度,分为 5 等：用整数 $1\sim5$ 表示,第 5 等最重要

他还从因特网上查到了每件物品的价格(都是 10 元的整数倍)

他希望在不超过 N 元(可以等于 N 元)的前提下,使每件物品的价格与重要度的乘积的总和最大

设第 j 件物品的价格为 v_j ,重要度为 w_j ,共选中了 k 件物品,编号依次为 $j_1,j_2,\cdots,j_k$,则所求的总和为：

$$v_{j1}\times w_{j1}+v_{j2}\times w_{j2}+\cdots+v_{jk}\times w_{jk}$$

请你帮助金明设计一个满足要求的购物单

数据规模

对于全部的数据 $1\le n\le32000,1\le m\le60,1\le v<10000,1\le p\le5$

输入格式

第一行,为两个正整数, N,m 其中 N 表示总钱数, m 为希望购买物品的个数

第 j 行给出了编号为 $j-1$ 的物品的基本数据,每行有 3 个非负整数 v,p,q

其中 v 表示该物品的价格, p 表示该物品的重要度, q 表示该物品是主件还是附件

如果 $q=0$,表示该物品为主件,如果 $q>0$,表示该物品为附件, q 是所属主件的编号

输出格式

只有一个正整数,为不超过总钱数的物品的价格与重要度乘积的总和的最大值

#1287、金明的预算方案

若直接按照上一题思路处理，时间复杂度 $O(mn^2)$

不难发现

- 依赖关系构成森林
- 每棵树的高度不超过 2 且为 2 叉树

对于一棵树若其附件数量为 k，其附件的选取状态一共有 2^k 种状态

不妨枚举每棵树的附件选取状态

将每种状态作为物品进行分组，每个组内有 2^k 件物品

保证组内仅允许选取一件物品，分组背包处理即可

时间复杂度 $O(nm2^k)$

```
for (int i = 1; i <= n; i++)
{
  scanf("%d%d%d", &w, &v, &fa);
  if (!fa)
    master[i] = {w, v * w};
  else
    slave[fa].push_back({w, v * w});
}
for (int i = 1; i <= n; i++)
  for (int j = m; j >= 0; j--)
    for (int k = 0; k < (1 << slave[i].size()); k++)
    {
      int w = master[i].w, v = master[i].v; //主件必须选取
      for (int u = 0; u < slave[i].size(); u++)
        if (k >> u & 1)
          w += slave[i][u].w, v += slave[i][u].v;
      if (j >= w)
        dp[j] = max(dp[j], dp[j - w] + v);
    }
printf("%d", dp[m]);
```

#745、树的深度和

题目描述

给定一个 n 个点的树

请求出一个结点,使得以这个结点为根时,所有结点的深度之和最大

一个结点的深度之定义为该节点到根的简单路径上边的数量

输入格式

第一行有一个整数，表示树的结点个数 n

接下来 $n-1$ 行

每行两个整数 u,v ,表示存在一条连接 u,v 的边

输出格式

输出一行一个整数表示你选择的结点编号

如果有多个点符合要求,输出编号最小的点

数据规模

对于全部的测试点,保证 $1 \leq n \leq 10^6, 1 \leq u,v \leq n$,给出的是一棵树

对于一棵无根树，不同节点作为根

深度和将会发生变化

枚举根并统计，时间复杂度 $O(n^2)$

#745、树的深度和

设 $\mathrm{dp}[u]$ 为以 u 为根节点时的深度和

不妨以 1 作为根进行一次 DFS

求出各节点深度 dep_u 及子树大小 cnt_u

观察右图，若根节点从 1 变为 2

有 cnt_2 个节点深度减少 1，$n - \mathrm{cnt}_2$ 个节点深度增加 1

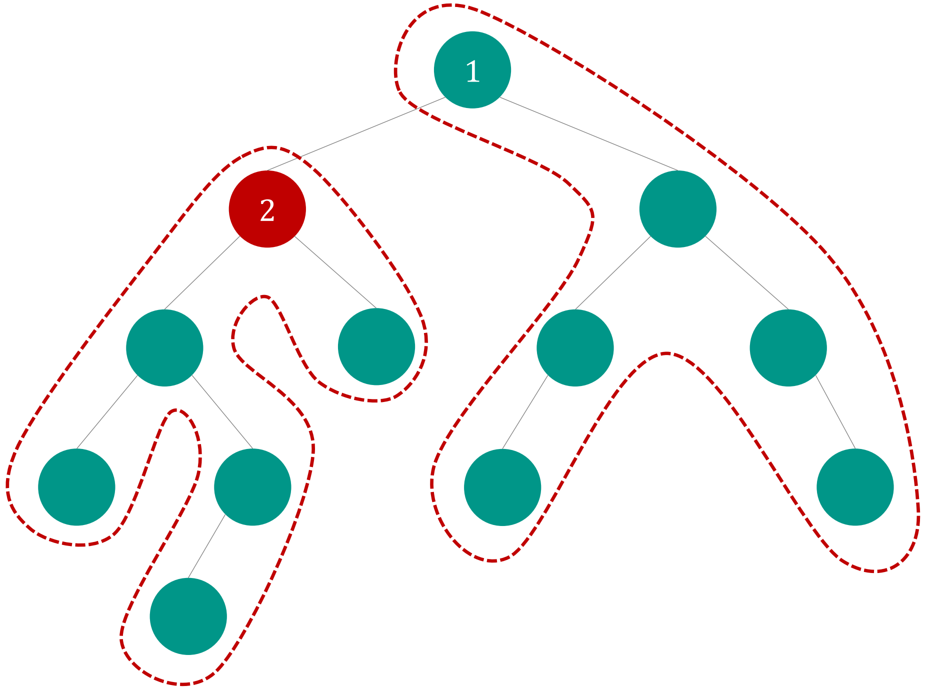
对于右图有 $\mathrm{dp}[2] = \mathrm{dp}[1] - 2 \times cnt_2 + n$

再进行一次 dfs，有如下递推关系

$$\mathrm{dp}[u] = \mathrm{dp}[fa_u] - 2 \times \mathrm{cnt}_u + n$$

时间复杂度 $O(n)$

该种处理方式被称作二次扫描（换根 DP）



#2865、树的中心

题目描述

给定一棵树

树中包含 n 个结点(编号 $1 \sim n$)和 $n-1$ 条无向边
每条边都有一个权值

请你在树中找到一个点,使得该点到树中其他结点的最远距离最近

输入格式

第一行包含整数 n

接下来 $n-1$ 行,每行包含三个整数 u_i, v_i, w_i

表示点 u_i 和 v_i 之间存在一条权值为 w_i 的边

输出格式

输出一个整数,表示所求点到树中其他结点的最远距离

数据范围

对于全部的数据 $1 \leq n \leq 2 \times 10^5, 1 \leq u_i, v_i \leq n, 1 \leq w_i \leq 1000$

可 Floyd 求出多源最短路，时间复杂度 $O(n^3)$

#1819 仅可求出经过各点且终点在其子树内的最大路径

若树型态变化可能产生新的子树

枚举各点作为根节点，参考 #1819 进行 DP

时间复杂度 $O(n^2)$

先 DFS 求出各点向下的最大距离 $\mathrm{d1}_u$ 和次大距离 $\mathrm{d2}_u$

同时记录 $\mathrm{d1}_u$ 除 u 外第一个节点编号 idx_u

设 up_u 为 u 先到达 fa_u 的最大距离

#2865、树的中心

对于 $v \in \mathrm{son}_u$ 在遍历到 u 时更新 v 的最大距离

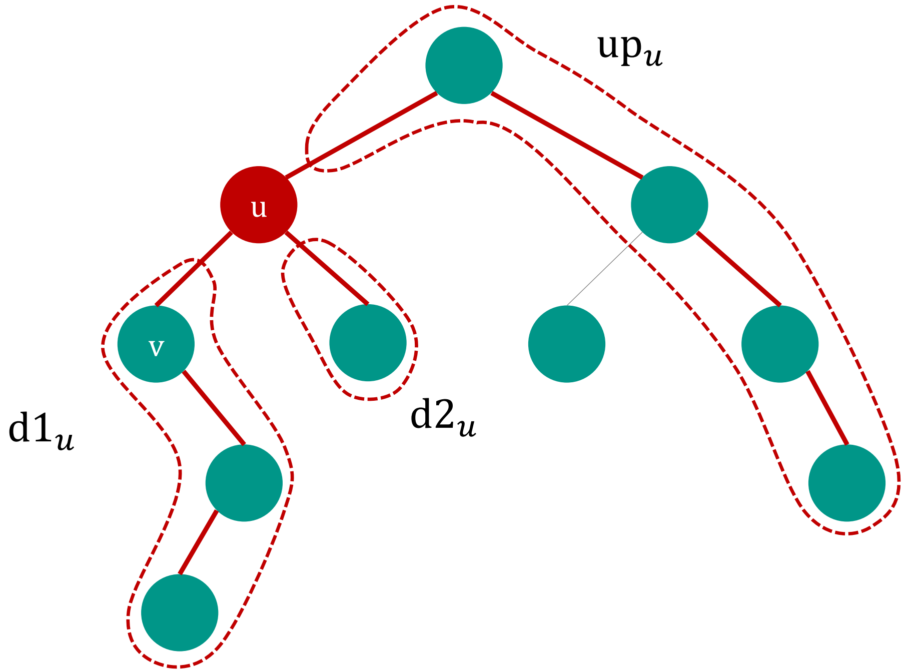
点 v 的最大距离有两类情况

- 向下走的最大距离 $\mathrm{d1}_v$
- 向上走，必经过其父亲 u
    - 其父亲 u 向上走，即 $\mathrm{up}_u + w_{u,v}$
    - 其父亲 u 向下走
        - 若 $\mathrm{idx}_u \neq v$，即 $\mathrm{d1}_u + w_{u,v}$
        - 若 $\mathrm{idx}_u = v$，即 $\mathrm{d2}_u + w_{u,v}$

再进行一次 DFS 求出所有 up_u（需先更新 up_v 再处理 v）

答案为 $\min\{\max(\mathrm{up}_u, \mathrm{d1}_u)\}$

时间复杂度 $O(n)$

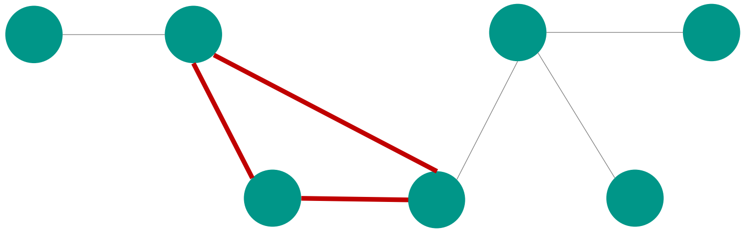


基环树

若无向连通图包含恰好一个环，则称它是一棵 **基环树**（pseudotree）

若有向弱连通图每个点的入度都为 1，则称它是一棵 **基环外向树**

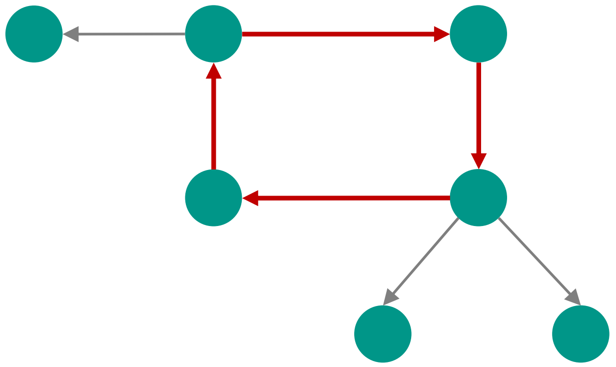
若有向弱连通图每个点的出度都为 1，则称它是一棵 **基环内向树**

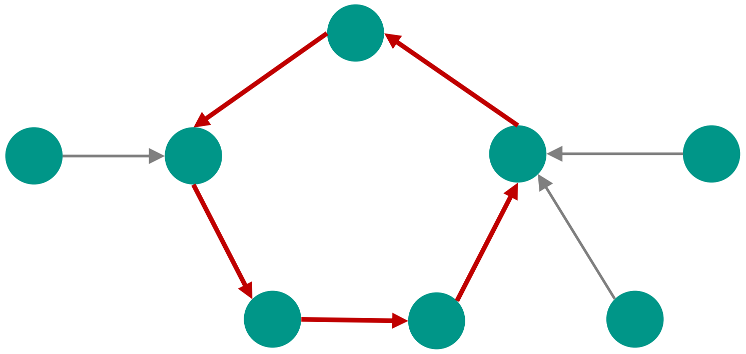


多棵树可以组成森林， 多棵基环树可以组成 **基环森林** (pseudoforest)

多棵基环外向树可以组成 **基环外向森林**

多棵基环内向树可以组成 **基环内向森林**





#2300、发现环

问题描述

SYC 有 N 台电脑,编号 $1 \sim N$.

原本这 N 台电脑之间有 $N-1$ 条数据链接相连,恰好构成一个树形网络

在树形网络上,任意两台电脑之间有唯一的路径相连

不过在最近一次维护网络时,管理员 Mas 误操作使得某两台电脑之间增加了一条数据链接,于是网络中出现了环路

环路上的电脑由于两两之间不再是只有一条路径,使得这些电脑上的数据传输出现了 BUG

为了恢复正常传输, Mas 需要找到所有在环路上的电脑,你能帮助他吗?

输入格式

第一行包含一个整数 N

以下 N 行每行两个整数 a 和 b
表示 a 和 b 之间有一条数据链接相连

输入保证合法

输出格式

按从小到大的顺序输出在环路上的电脑的编号,中间由一个空格分隔

可直接拓扑排序

对于无向图 ，不难发现

若为环上点其度必然不小于 2

时间复杂度 $O(n)$

数据规模

对于 30% 的数据, $1 \leq N \leq 1000$

对于 100% 的数据, $1 \leq N \leq 100000, 1 \leq a \leq b \leq N$

#2300、发现环

也可直接 DFS

不妨给各点打上时间戳 dfn，同时记录各点前驱 fa

DFS 时 当前点 u 下一节点 v

- 若 $\mathrm{dfn}_v = 0$

  说明未访问直接访问

- 若 $\mathrm{dfn}_v > \mathrm{dfn}_u$

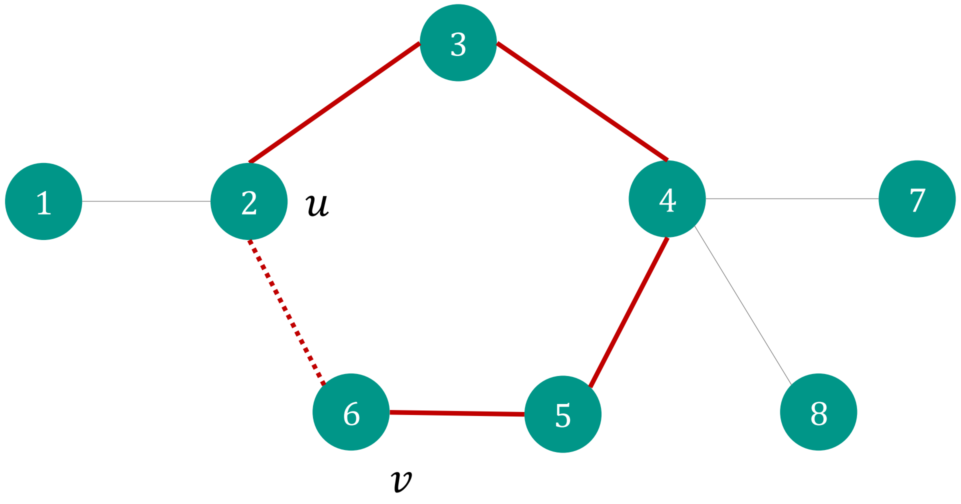
  说明 $u \leftrightarrow v$ 为一条环上边

  不断令 $v \leftarrow fa_v$ 直到 $v = u$

  同时 v 及 将上跳的各点记录，这些点都为环上点

- 否则不做处理

时间复杂度 $O(n)$



#747、骑士

题目描述

Z 国的骑士团是一个很有势力的组织,帮会中聚集了来自各地的精英

他们劫富济贫,惩恶扬善,受到了社会各界的赞扬

可是,最近发生了一件很可怕的事情：邪恶的 Y 国发起了一场针对 Z 国的侵略战争

战火绵延五百里,在和平环境中安逸了数百年的 Z 国又怎能抵挡得住 Y 国的军队

于是人们把所有希望都寄托在了骑士团身上,就像期待有一个真龙天子的降生,带领正义打败邪恶

骑士团是肯定具备打败邪恶势力的能力的,但是骑士们互相之间往往有一些矛盾

每个骑士有且仅有一个他自己最厌恶的骑士(当然不是他自己),他是绝对不会与最厌恶的人一同出征的

战火绵延,人们生灵涂炭,组织起一个骑士军团加入战斗刻不容缓！国王交给你了一个艰巨的任务：
从所有骑士中选出一个骑士军团,使得军内没有矛盾的两人,即不存在一个骑士与他最痛恨的人一同被选入骑士团的情况,并且使这支骑士军团最富有战斗力

为描述战斗力,我们将骑士按照 $1 \sim N$ 编号,给每位骑士估计一个战斗力,一个军团的战斗力为所有骑士的战斗力之和

输入格式

输入第一行包含一个正整数 N ,描述骑士团的人数

接下来 N 行每行两个正整数,按顺序描述每一名骑士的战斗力和他最痛恨的骑士

输出格式

输出一个整数,表示你所选出的骑士军团的战斗力

数据范围

对于 30% 的数据,满足 $N \leq 10$

对于 60% 的数据,满足 $N \leq 100$

对于 80% 的数据,满足 $N \leq 10^4$

对于 100% 的数据,满足 $N \leq 10^6$,且每名骑士的战斗力都是不大于 10^6 的正整数

若只有 $n-1$ 条边

该题解法同 #2861

#747、骑士

若有 n 条边将构成 **基环森林**

将每个连通块中的环找出，对于环上的任意一条边 $u \leftrightarrow v$ 将其断开

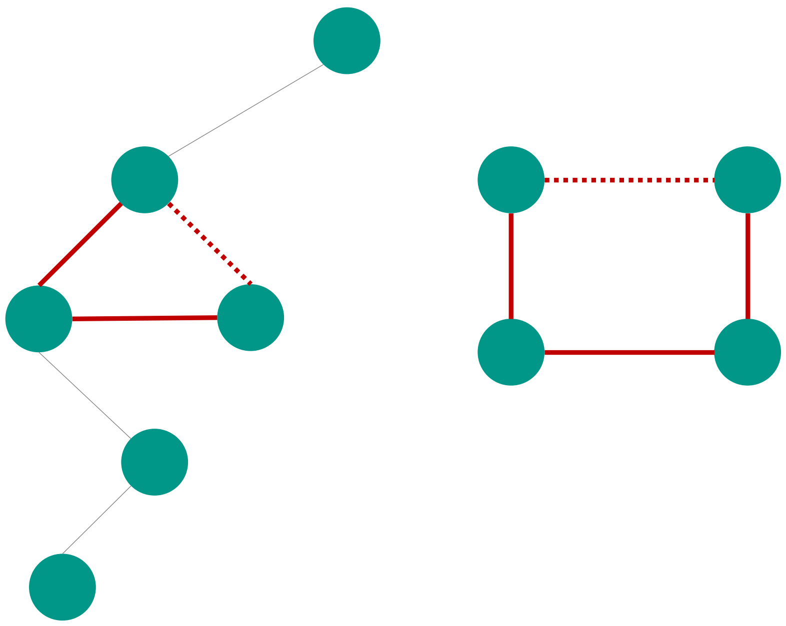
那么对于该连通块不再有环，必然为一颗树

参考 #2861

- 将 u 做为根求出 u **不选时**的最大攻击力
- 将 v 做为根求出 v **不选时**的最大攻击力

对于每个连通块 C 累加 $\max\limits_{u \in C} (\mathrm{dp}[u][0])$ 即可

时间复杂度 $O(n)$



u, v 中选取一个时的方案，若其为最大值必然被包含在另一点不选的方案中

不需考虑断开环上所有边，若环上其它点不选时成为最优解，其必然也被包含在 $u \leftrightarrow v$ 断开的方案中

LAB
实验舱
青少年编程
走近科学 走进名校

谢谢观看