# 树形动规总结

**树型动规的基本方式同普通的线性动态规划，但遍历的顺序是由高深度向低深度直至根节点，通常一个树型动规包括了状态、阶段、决策、状态转移方程，找到每个题目对应的动归要素，是一个题目的难点所在。**

1. **状态：程序求解到每个程度所储存的信息，通常我们需要在一开始找到适合题目的状态并进行定义。Tips：找状态可以通过确定变量，权衡变量的范围来寻找。**
2. **阶段：就是求解动态规划的顺序，每次求解的状态都必须运用之前已经求解过的作为辅助。**
3. **决策：选择最优解的过程，通常是求最大值，最小值等。**
4. **状态转移方程：思考出决策后，用状态转移方程将其表达出来。**

**关于求解顺序：由于树形动归的特殊性（几乎都是无向边），我们应该从叶子节点开始遍历，有三种方法：**

1. **先给树做一次BFS，然后将队列中的元素一一出队，就是树型动规的顺序。**
2. **找度为2的节点，记录，并且更新它的子节点的度数，重复这个拓扑排序操作，得到的也是树型动规的顺序。**
3. **用递归将树的后序遍历求出，即可将一棵树线性化**

**拓展：状态压缩**

**当我们的某个状态过于繁琐，很难将其用一个维度表示出来，怎么办？**

**如山贼集团题目，设立分部的时候各个分部会互相影响，而且不同的分部之间影响的情况是不同的，不像我们通常的动归，所有的物品一视同仁。**

**比如某结点选择1、2、3分部，会导致总资金损失，而选择1、4分部，总资金却会增加……**

**观察分部的总数量，小于等于8，这个时候我们可以用到位运算的相关知识，将1~8分部的选择情况用一个int变量储存起来，从而起到表示状态的作用。**

**Ural 1039 没有上司的晚会**

背景

有个公司要举行一场晚会。

为了能玩得开心，公司领导决定：如果邀请了某个人，那么一定不会邀请他的上司

（上司的上司，上司的上司的上司……都可以邀请）。

题目

每个参加晚会的人都能为晚会增添一些气氛，求一个邀请方案，使气氛值的和最大。

输入格式

第1行一个整数N（1<=N<=6000）表示公司的人数。

接下来N行每行一个整数。第i行的数表示第i个人的气氛值x(-128<=x<=127)。

接下来每行两个整数L，K。表示第K个人是第L个人的上司。

输入以0 0结束。

输出格式

一个数，最大的气氛值和。

样例输入

7

1

1

1

1

1

1

1

1 3

2 3

6 4

7 4

4 5

3 5

0 0

样例输出

5

**提交地址：**

<http://acm.timus.ru/submit.aspx?space=1&num=1039>

**首先定义状态：每个结点，然后我们发现在一个人在不参与聚会时，它的相邻子节点可以参加聚会，也可以不参加；一个人参与聚会时，它的相邻子节点一定不能参与聚会。我们加一个状态：参与或者不参与聚会。**

**dp[i][j]表示第i个人参与(j==1) 或不参与(j==0)聚会时聚会所能达到的最大气氛值**

POJ 1985 Cow Marathon

题目大意

求一棵树的最长路径

输入：

第一行：n,m (2 <= N <= 40,000,1 <= M < 40,000) ，表示有n个节点，m条边

接下来m行：

每行四个量：a b w c，表示a与b之间有一条权值为w的路径，将c忽略掉

输出：

最长路径长度

**模版题**

**Ural 1018 二\*苹果树**

**题目**

有一棵苹果树，如果树枝有分叉，一定是分2叉（就是说没有只有1个儿子的结点）  
这棵树共有N个结点（叶子点或者树枝分叉点），编号为1-N,树根编号一定是1。  
我们用一根树枝两端连接的结点的编号来描述一根树枝的位置。下面是一颗有4个树枝的树  
   2   5  
    \ /   
     3   4  
      \ /  
       1  
现在这颗树枝条太多了，需要剪枝。但是一些树枝上长有苹果。  
给定需要保留的树枝数量，求出最多能留住多少苹果。

**输入格式**

第1行2个数，N和Q(1<=Q<= N,1<N<=100)。  
N表示树的结点数，Q表示要保留的树枝数量。接下来N-1行描述树枝的信息。  
每行3个整数，前两个是它连接的结点的编号。第3个数是这根树枝上苹果的数量。  
每根树枝上的苹果不超过30000个。

**输出格式**

一个数，最多能留住的苹果的数量。

**样例输入**

5 2  
1 3 1  
1 4 10  
2 3 20  
3 5 20

**样例输出**

21

**状态：结点、树所保留的树枝数量。**

**f[i][j]表示在第i个结点以下的子树保留j个树枝所能保留的最大苹果数**

**分配树枝需要枚举分配给整个子树的树枝数、再枚举分给左右子树的树枝各有多少，注意这里分给左右子树树枝的同时会损失掉两根树枝**

**POJ1155 有限电视网络**

有一棵N个节点的树，树上有M个叶子节点，对应M个用户，其余为转发站，1号节点为根，电视台在1号节点，节目从一个地方传到另一个地方都要费用，同时每一个用户愿意出相应的钱来收看节目。求在电视台不亏本的前提下，最多允许有多少个用户可以看到电视节目。

规模：

N<=3000 M<N

输入：

N M N表示转发站和用户总数，M为用户数

以下N-M行，第i行第一个K，表示转发站i和K个（转发站或用户）相连, 其后第j对数val,cost表示，第i个转发站到val有边，费用cost.

最后一行M个数表示每个用户愿意负的钱。

输出：

不亏本前提下，可以收到节目最多的用户数。

（如果某个用户要收到节目（叶子结点），那么电视台到该用户的路径节点的费用都要付）

思路:在树上进行背包，对于以u为根的子树,该子树供给给j个用户亏本的最少钱

**此题用平常的思维一般会想到用f[i][j]的i表示结点，j表示盈亏费用，f[i][j]就刚好表示用户的数量，但是盈亏费用并不是一个很好的量，它的范围很大，即使用动态规划也会很耗时间，所以我们只能将f[i,j]表示在以i为根的树上允许j的用户数，最大能赚到（最少亏损）的钱**

**在最后全部遍历一边，最先被发现f[i][j]>=0的i就是答案**

**另外还有一个问题，一个根结点有多个子节点，不像二叉苹果树那么容易枚举出来，这里可以用到一些01背包的思想，如果用动态规划来求解这个动态规划的状态转移方程，将每个子节点看作物品，分配的则是对应的用户数，能在较快的时间内求出状态**

# 山贼集团

时间限制：4s 空间限制：256MB

题目描述

某山贼集团在绿荫村拥有强大的势力，整个绿荫村由N个连通的小村落组成，并且保证对于每两个小村落有且仅有一条简单路径相连。小村落用阿拉伯数字编号为1,2,3,4,…,n，山贼集团的总部设在编号为1的小村落中。山贼集团除了老大坐镇总部以外，其他的P个部门希望在村落的其他地方建立分部。P个分部可以在同一个小村落中建设，也可以分别建设在不同的小村落中。每个分部到总部的路径称为这个部门的管辖范围，于是这P个分部的管辖范围可能重叠，或者完全相同。在不同的村落建设不同的分部需要花费不同的费用。每个部门可能对他的管辖范围内的小村落收取保护费，但是不同的分部如果对同一小村落同时收取保护费，他们之间可能发生矛盾，从而损失一部分的利益，他们也可能相互合作，从而获取更多的利益。现在请你编写一个程序，确定P个分部的位置，使得山贼集团能够获得最大的收益。

输入格式(cateran.in)

输入文件第一行包含一个整数N和P，表示绿荫村小村落的数量以及山贼集团的部门数量。

接下来N-1行每行包含两个整数X和Y，表示编号为X的村落与编号为Y的村落之间有一条道路相连。(1<=X,Y<=N)

接下来N行，每行P个正整数，第i行第j个数表示在第i个村落建设第j个部门的分部的花费Aij。

然后有一个正整数T，表示下面有T行关于山贼集团的分部门相互影响的代价。(0<=T<=2p)

最后有T行，每行最开始有一个数V，如果V为正，表示会获得额外的收益，如果V为负，则表示会损失一定的收益。然后有一个正整数C，表示本描述涉及的分部的数量，接下来有C个数，Xi，为分部门的编号(Xi不能相同)。表示如果C个分部Xi同时管辖某个小村落（可能同时存在其他分部也管辖这个小村落），可能获得的额外收益或者损失的收益为的|V|。T行中可能存在一些相同的Xi集合，表示同时存在几种收益或者损失。

输出格式(cateran.out)

输出文件要求第一行包含一个数Ans，表示山贼集团设置所有分部后能够获得的最大收益。

样例数据

|  |  |
| --- | --- |
| 输入样例 | 输出样例 |
| 2 1  1 2  2  1  1  3 1 1 | 5 |

数据规模

对于40%的数据，1<=P<=6。

对于100%的数据，1<=N<=100，1<=P<=12，保证答案的绝对值不超过108。

**题目并不难，难在储存各个不同分部的分配方法上，前文讲到可以使用状态压缩的方式，这里给出一种具体的实现方法：**

**for(int j’= j ;j’>=0;j’=(j’-1)&j){**

**}**

**其中的j’就是j方案的一个补集，反复循环，直到遍历到空集才结束**

**如00010011 表示选4、7、8分部**

**则它的补集为：**

**S1=00010010&00010011=00010010 4、7分部**

**S2=00010001&00010011=00010001 4、8分部**

**S3=00010000&00010011=00010000 4分部**

**S4=00001111&00010011=00000011 7、8分部**

**S5=00000010&00010011=00000010 7分部**

**S6=00000001&00010011=00000001 8分部**

**S7=00000000&00010011=0**