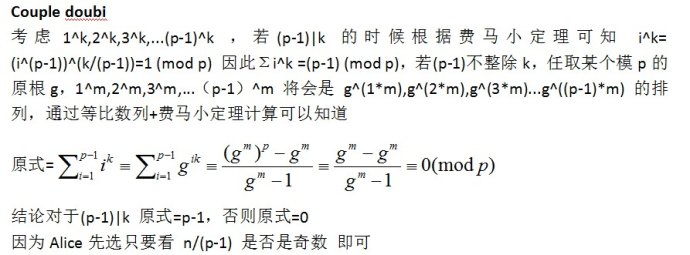
2014 Multi-University Training Contest 1--by FZU 解题报告

**A.**

[](http://photo.blog.sina.com.cn/showpic.html#blogid=6bddecdc0102uy9g&url=http://album.sina.com.cn/pic/001YtlfKty6KEeGv5cofc&690)

**B. Jump**

最小K路径覆盖的模型，用费用流或者KM算法解决，构造二部图，X部有N\*M个节点，源点向X部每个节点连一条边，流量1，费用0，Y部有N\*M个节点，每个节点向汇点连一条边，流量1，费用0，如果X部的节点x可以在一步之内到达Y部的节点y，那么就连边x->y，费用为从x格子到y格子的花费能量减去得到的能量，流量1，再在X部增加一个新的节点，表示可以从任意节点出发K次，源点向其连边，费用0，流量K，这个点向Y部每个点连边，费用0，流量1，最这个图跑最小费用最大流，如果满流就是存在解，反之不存在，最小费用的相反数就是可以获得的最大能量

**C. Centroid of a Tree**

首先找出重心，之后以重心为根进行树形dp，用dp[i][j]表示以编号为i的节点为根的子树有j个儿子的方案数 ，这个直接合并，每次合并两个子树的时候枚举两个子树的儿子的个数，对于树上每条边合并一次，总共合并n 次，每次合并的复杂度是n^2，这个是三方的复杂度。

对于有两个重心的情况，首先将这两个子树中间的边打断变成两个树，对这两个树分别按照上述方法处理出dp 数组，如果要使得重心位置不变，那么这两个子树需要有相同的节点个数，这个直接枚举一下然后两边乘起来 就好了。

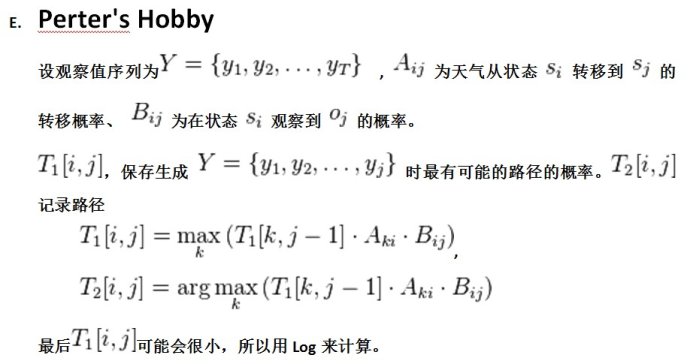
至于只有一个重心的情况，还是按照上述方法处理出dp数组，为了计算合法的方案数，我们用总的方案数减去 不合法的方案数，对于不合法的情况，去掉这个根后一定会有一个最大的分支，这个分支的节点数超过剩下的 全部分支加起来的节点个数，所以先枚举最大分支，对于剩下的分支用背包处理，f[i]表示剩下的全部子树取i 个节点的方案数，这个很容易处理，处理完之后枚举一下最大分支的节点个数，再利用f数组就可以算出在这种 情况下的非法的方案数了。

对于这里的复杂度要特别说明一下，设每个分支的节点数是x1,x2,x3...,xk，合并的复杂度是x1\*x2+(x1+x2) \*x3+...+(x1+x2+...+xk-1)\*xk<=(x1+x2+...+xk)^2=n^2，加上枚举是三方，所以整个dp的总复杂度是三方的。

**D. Task**

基本思想是贪心。

对于价值c=500\*xi+2\*yi，yi最大影响100\*2<500,所以就是求xi总和最大。可以先对机器和任务的时间从大到小排序。从最大时间的任务开始，找出满足任务时间要求的所有机器，从中找出等级最低且满足任务等级要求的机器匹配。依次对任务寻找满足要求的机器。

[](http://photo.blog.sina.com.cn/showpic.html#blogid=6bddecdc0102uy9g&url=http://album.sina.com.cn/pic/001YtlfKty6KEeGvFkt81&690)

**F. Shooting**

将所有目标与起点线的距离离散化作为下标，建立函数式线段树，将距离按区间端点从1到X的顺序加入函数式线段树，左端点+1,右端点-1，记录区间元素的距离和，以及元素的个数。对于在x位置的询问，找到其对应的端点，这个可以二分找到，然后在该端点对应的线段树上进行二分查找求解，最后判断前一个答案与P的大小得到当前问题的答案。

**G. Xor**

线段树，时间复杂度是O（5000\*200\*logn+10\*m\*logn）。

f[ prefix ]保存2进制前缀状态，合并就暴力，比如

101xxx

11xxxx

x表示0或者1  
前缀固定 后面随便取  
这2个xor的结果  
只有前2位是确定的

合并完大约就是

01xxxx

然后方案数就可以算了，可以证明，状态不会太多。

**H. Information Extraction**

本题要求大家能够存储特定的输出格式以及若干种HTML结构和映射关系，根据输入的HTML文本，寻找其结构，如果找不到结构与之匹配，输出“Can't Identify”，如果只有一个，根据映射关系以及输出格式的要求输出结果，如果有多个，则使用最早输入的那个结构。

解决以下几点即可

1、对HTML输入的读取

2、判断结构是不是输入的HTML文本的结构

3、根据映射以及输出格式的要求输出结果

注意映射关系中可能有一个id到多个标签，并且这些标签在输入格式中是存在多个的。

**I. Turn the pokers**

最终的结果一定是连续出现的，只需要求出最终的区间。

因为如果对同一张牌进行两次操作，牌的状态不改变。故牌的翻转次数一定是减少偶数次。如果所有数的和是奇数，那么最终结果也一定是奇数。同理，偶数也是一样的。

所以只要递推求出最后的区间，计算sum（C（xi，m）（i=0，1，2。。。）），m是总牌数，xi是在区间内连续的奇数或偶数，在模10^9+9就是最终的答案。

**J. Rating**

令(x, y)表示高分为x，低分为y的状态（x >= y），E(x, y)表示从(x, y)到达(1000, ?)的比赛场数期望。容易得到E(x, y) = P \* E(x1, y1) + (1 - P) \* E(x2, y2) + 1，其中，(x1, y1)表示rating上升后的状态，（x2, y2）表示rating下降后的状态。把E(1000, ?) = 0带入可以得到包含n个未知数的n个方程，n大概200多，可以高斯消元。E(0, 0)即为答案。

**K. Shortest-path tree**

首先构造最短路径树。先求根节点到其他节点的最短路径，然后从根节点开始进行深度优先遍历，先遍历节点编号较小的没遍历过的儿子，这样就能处理处最短路径树。

之后找节点数为K的树链。可以用树分治进行求解，在树分治求解过程中，对于每个中心点，处理出该子树中所有节点到中心点的树链，然后枚举每条树链，比如某条树链节点为a，长度为b，则找之前遍历过的树链中节点数位K-a的长度最长的树链，将这两条树链拼起来可以得到节点数为K的树链，用其更新答案。最好一个一个分支分别处理，可以避免考虑同一个分支来的两条树链。这样枚举并且更新信息，还要存方案数。