Comet OJ - Contest #12

A. Competitive Problem Setting

这个题想法来自于最近做的一些波兰题，众所周知波兰人的题目都有一个长度为3的编码，第一个样例其实就是PA 2010题目的标题。做法很简单，用一个`set`存一下所有字符串的长度为3的前缀，然后看看`set`的大小是否个字符串个数一样即可。

出题人代码

Challenge：如果长度为3的前缀和后缀都可以当做编码，且总字符串个数有10^6个，应该怎么做？

B. Binary Matrix Transform

先考虑最简单的情况，仅有c操作且$c=1$。那么显然，只要数一数两个矩形每一行里面`1`的个数的奇偶性是否相同。

接下来考虑r和c操作同时存在，且$r=c=1$。这个稍微分析下，也可以发现只要数一数两个矩形里面`1`的个数的奇偶性是否相同即可。

对于$r=2$或者$c=2$，我们可以发现只要把奇数列，偶数列，奇数行，偶数行，拆分出来考虑，每个拆出来的小矩形都是一个$r=c=1$的问题。

出题人代码

Challenge：如果其中一个操作变成给出一个$p\times q$的`01`矩阵，可以选A的一个$p\times q$的连续子矩阵异或下，应该怎么做？

C. Bus Station

考虑构建这样一个有向无环图，每个点对应了题目中的一个时刻。考虑一班公交，如果t_i和t_j ($t_i<t_j$)时刻是在相邻两个公交站，那么可以从t_i连一条长度为0的边到t_j。对于同一个公交站，把经过这个站的公交的入站时间从小到大排序，那么相邻两个时刻t_i可以连一条长度为t_j-t_i的边到t_j。

最后，我们只要求出一条从t_1到t_2的最短路即可。

出题人代码

D. XOR Pair

首先不考虑x和y的范围限制，以及$x-y$不搞任何借位，那么$x\oplus y=n$，其实限制了$x-y$的二进制表示中某一位是-1，0或者1。特别地，如果n二进制表示的第i位是0，那么$x-y$的二进制表示的第i位也一定是0。如果n二进制表示的第i位是1，那么$x-y$的二进制表示的第i位可以是-1或者1（取决于x二进制表示的第i位是0还是1）。

那么不妨考虑把那些可以是-1或者1的位全部当做-1，那么你把第i位搞成1的时候等价于加上2^{i+1}。问题转化成，你开始有一个数s（其实$s=-n$），然后你可以选择一些i，给s加上2^{i+1}，使得最终这

个数的绝对值不超过m。再转化下，其实就是选出来的那些i构成的数要在区间$[n-m, m+n]$里面。

有了这样一个转化，我们可以用数位dp来做这个题了。令$ways(i, ex, ey, el, er)$表示从高到低处理到了第i位，x和a的大小关系是ex(1表示相同，0表示小于)，y和b的大小关系是ey(1表示相同，0表示小于)，最终选出来那个数和$n-m$以及$n+m$的大小关系分别是el和er。转移过程就是枚举下x和y的第i位分别取什么即可。

出题人代码

E. Ternary String Counting

做过今年HDU多校的人可能知道有这样一个题Blank，官方做法是个$O(n^4)$的dp。经过研究发现，可以优化到$O(n^3)$，于是就出了这个题。

考虑朴素的$O(n^3)$的做法，$ways(i, j, k)$表示填上了前i位后，最近的两个不同字符的位置分别在j和k $(i > j > k)$的方案数。题目中的限制条件其实给j和k限制了一个范围$[\text{jmin}_i, \text{jmax}_i]$和$[\text{kmin}_i, \text{kmax}_i]$。在做转移的时候，对每个在范围内的$j$和$k$ ($\text{jmin}_i \le j \le \text{jmax}_i, \text{kmin}_i \le k \le \text{kmax}_i, k < j$)，其实都有如下的几种转移：

1. $ways(i+1, i, k) \overset{+}{\leftarrow} ways(i, j, k)$
2. $ways(i+1, i, j) \overset{+}{\leftarrow} ways(i, j, k)$
3. $ways(i+1, j, k) \overset{+}{\leftarrow} ways(i, j, k)$

很容易发现，第一维完全是没有用处的，因为在前两个转移中，我们的第二维总是会变成i，在第三个转移中，j和k完全没有变化。因此这个dp的转移其实等价于，每次给出一个矩形，先把矩形外的值全部清零。然后前两种转移对应了对某一行求和，或者某一列求和，第三种转移不改变dp数组的值。然后还可以发现，一旦某个值被清零，那么这个值以后永远都是零。并且对于某一行来说，非零的值永远是一段连续区间。

那么，我们可以维护这个dp数组第i行非零区间的左端点和右端点，每一行的和以及每一列的和。每次来一个矩形，先对每一行清零，清零过程中维护每行的和以及每列的和。

时空复杂度$O(n^2)$。

出题人代码

F. Substring Query

不妨考虑$i - 1 \ge n - j$，也就是左半边长度大于右半边的情况。其他情况只需要把串反过来做一遍即可。显然，half border分成了两部分：

1. 长度不超过$n-j$，这个我们仅需要求出原串的所有border，按照长度排序，求前缀和，然后二分下即可。
2. 长度超过$n-j$，这个比较麻烦，但是可以发现这种half border一定在左半边内，不会超过，接下来考虑如何快速统计第二种。

令左半边的字符串是x，右半边是y，那么我们就是要求出全部的字符串z，使得zy是x的一个前缀。显然z是x的一个border，y是x的一个子串且这个子串前面恰好是z。我们需要想办法求出所有z和y的

交界处。

令prefix_i表示长度为i的前缀的最长border的长度，令suffix_i表示后缀$s_{i..n}$的最长border的长度。考虑构建这样两棵有根树树：

1. i的父节点是prefix_i
2. i的父节点是$n-\text{suffix}_i+1$

对于第一棵树，我们知道长度为i的前缀的border在0到i这条路径上。对于第二棵树，我们可以知道i子树中的所有节点，都以$s_{i..n}$为前缀。显然，对于一个x和y，所有交界处显然就是0到x在第一棵树上的路径和y在第二棵树上的子树的交集。

那么对于一个询问(i,j)，我们其实就是要在第一棵树上0到$i-1$的路径上找一些点k，使得$k+1$在第二棵树上是$j+1$的子孙。并且要满足$2(k+n-j)\le i-1+n-j$（half border）。这个离线后就是一个简单的二维数点问题。遍历第一棵树，用一个树状数组维护当前路径上点在第二棵树上的标号。询问的话就是个树状数组区间和。

出题人代码

Challenge：如果改成询问所有border的和应该怎么做？