——不变量与等价关系

谈问题的拓扑结构

毫无用处的知识

- 本意：给定全集S和其子集族T，如果：
  - 空集Ø和全集S属于T
  - T中两个（有限个）元素之交属于T
  - T中任意多（有限或无限个）元素之并属于T
- 那么T称为S的拓扑。(S, T)称为拓扑空间。
- 拓扑空间A到B的一一的、连续的映射，如果其逆映射也是连续的，那么两个空间拓扑同胚。
- 同胚的空间**没有实质性的不同**。

问题的拓扑性质

- 当然我们通常使用“拓扑”这个词的时候都不是本来的意思。但是这给我们启发……
- 通常的问题是，有一系列解，有的可行，有的不可行。我们要找全体可行解中的一个整体的性质。比如说求个数，最优解，期望等等。
- 如果我们有一个等价关系，能够保证可行解和不可行解不会相等，那么我们可以利用这个等价关系缩小解集的规模。

例1 网教“小学数学竞赛题1”

- 给定一个2Nx2N的棋盘，放入若干个棋子。
- 先任意选取N行，取走其上全部的棋子；再任意选取N列取。
- 如果不管选择怎样的方案取都无法取完所有的棋子，那么这种放棋子的方案是好的。
- 要得到一个好方案，最少要放多少颗棋子？在此基础上，有多少种不同的放法？答案模P。

例1

- 2N个点？每行每列一个，显然不够。
- “每有一行有2个点，就消那一行。”按这个策略，3N个点也可以完全消去。
- 3N+1个点？似乎可行。
- 试试2N=2,4,6时，都有解。
- 一共N+1行/列有2个，其他行/列有1个。
- 从这些行中任意选N行消去，剩下1行的两个元素所在列只有1个元素，外加上N-1个独立的行列，就是正解了。

例1

- 重要的是每一行、列的点，而不是和特定点相邻的点。
- 行列重排后，结果不会变。（该消哪行消哪行）
- 于是我们想办法找到一个“标准型”，代表乱七八糟的摆放方式。
- 首先可以把(1,1),(2,2),......(2N,2N)放好2N个，剩下的从1行开始放，放到N+1行。
- 于是我们可以用N+1个数组成的数列代表一类“换汤不换药”的放置方法。 =>等价类

例1

- 但是还不够……
- 譬如说，以2N=6为例。(1,5)和(1,2)有区别吗?
- 没有。把2行5行和2列5列分别交换即可。
- 于是整个序列就会是以下两种:
  - 2,3,4,5：显然这是非法解。（消前3行和后3列）
  - 中途出现了某一行放置的列小于行号。比如说2,3,4,2
- 对于后者，我们继续考虑……

例1

- 1->2，2->3，3->4，4->2?
- 画成有向简单图：一个1->2，加上一个环。
- 重要的不是这个序列具体数字，是链和环！
- 这个N+1个点的图中，每个点的出度和入度最多是1（否则那一行/列有3个点）
- 于是要么就是一个N+1元环，要么就是若干个小环的并。

例1

- 若干个小环的并是不行的！
- 比如说有一个x元环，x<=N，那么我们可以直接消去环所在的列，这样这个环就没有了。
- 剩下所有的环都按行消。
- 最后剩下的独立元素按行、列任意都可以消。

- 于是就证明了，全部的解其实是一种情况在行列重排下的变形！这种情况是:
- 2,3,4,……,N+1,1

例1

- 于是我们得到了所有的合法解。
- 个数？计算这一种能生成多少种不同方案就好了。
- 直接行列重排前(N+1)x(N+1)显然是有重复的。
- 我们发现我们可以继续利用那个有向图。在具有单环拓扑结构的基础上，有向图的种类数就是一个N+1的圆排列：N!

例1

- 然后不动列，进行行重排：(N+1)!
- 发现1->2->3->4->1和1->4->3->2->1是相等的，也就是说实际上是一个无向图：0.5
- 把剩下的(N-1)的独点插入，就是先插入(N-1)行再插入(N-1)列：$\left(C_{2N}^{N-1}\right)^2$
- 最后的答案是：$\frac{1}{2}N!\,(N+1)!\left(C_{2N}^{N-1}\right)^2$
- 总复杂度 $O\left(\sqrt{N}\log^2 N\right)$

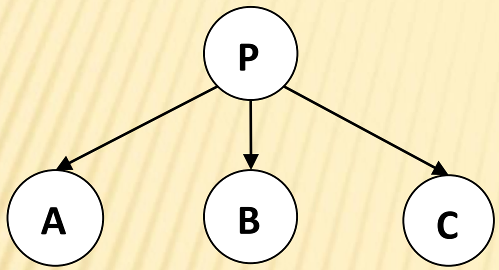
例2 UOJ122 树的计数

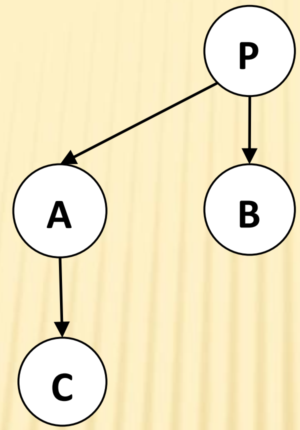
- 给出一棵树的DFS序和BFS序
- 这棵树的构造显然不唯一
- 求平均高度（NOI2013 Day1 2题）
- 求平均直径（2014年西安现场赛D题）

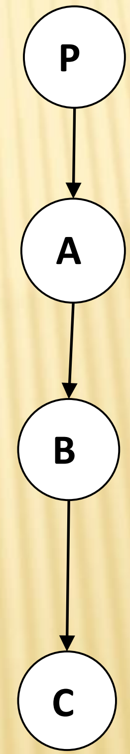
- 如果所有可能的树都是一种树通过某种变形得到的……
- 怎么可能有那么好的事！

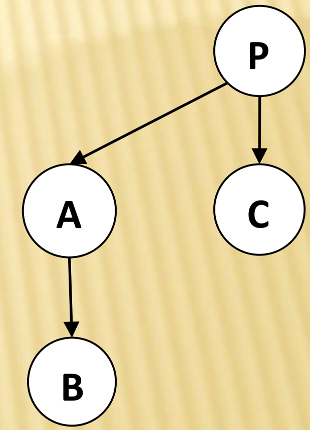
例2

- 先考虑高度好了。看下面四种：









- 2的DFS序和其他的不同，4的BFS序和其他的不同。
- 因此只有1和3是合法的不同情况。
- 总结：两序相邻的两个，既可为兄弟又可为父子。

例2

- 进一步分析两序都为PABC的情况。P和A显然只能是父子，其他AB和BC各自有2种情况。
- 画图后发现一共有4种情况，而且AB和BC互不干涉。不光在方案数上互不干涉，它们形成的高度也互不干涉。
- 直觉：分每一组这样的相邻关系讨论可以吗?

- 我们需要进一步的分析。

例2

- 出现偏差了怎么办?
- BFS: PABCD DFS: PACDB
- B只能做A的兄弟，C一定是A的孩子，D两可

- BFS: PABCD DFS: PACBD
- B只能做A的兄弟，C是孩子，D做C的兄弟

- BFS: PABCDE DFS: PACBDE
- B是A的兄弟，C是孩子，D是B的孩子，E两可

例2

- BFS: PABCDE   DFS: PADBCE
- B是A的兄弟，C也是兄弟，D是A的孩子，E是C的孩子

- BFS: 12345678 DFS: 12536748
- 234是一层，5是2的孩子所以在下一层(+1)，7两可……
- 等下，如果7是6的孩子不应该在8的后面吗?
- 答案是3，只有一种情况。

例2

- 假设BFS是顺序排列的，按BFS顺序一层一层地扫
- 为什么是BFS？因为BFS的层数和树的高度是一致的。
- 对BFS序中每一对相邻的点A和B，考虑其关系：
  - A是第一个点，那么B只能做A的儿子，答案加1。
  - B在A之前，那么B一定是A之前某个点的孩子，也就是说B在A的下一层，答案加1，开始下一层。
  - A和B在两序中都相邻。此时如果存在点C，使得C的BFS序比B大而DFS序比B小，那么C已经确定是A的下一层了，B是A的兄弟，答案不加；否则B为两可，本层答案加0.5。
  - DFS序中B在A之后但不相邻。此时DFS序中间夹的是A的孩子（BFS序大于B）和B的父亲（BFS序小于A）。无论哪种AB只能是兄弟。注意后者需要把当前层加的答案清理掉。

例2

- 一个值得注意的点是，所有的0.5都是在每一层的最右被加的，而且每一个不确定的点一定是当前层数最深的点。也就是说，树的形状是“几乎确定”的。
- 上述算法总复杂度O(N)。

- 其实把DFS排成1..N，按BFS扫，也是可以做的。
  - 当BFS[i]<BFS[i+1]时说明BFS[i+1]在下一层，答案加1。
  - 当BFS[i]+1==BFS[i+1]且BFS在i+1之后的全部n-i-1个数取遍数轴上连续的一段的时候，BFS[i+1]是两可的点。
  - 否则BFS[i+1]只能做BFS[i]的兄弟。
- 请自行思考原因。

例2

- 然后我们再来讨论直径的问题。

- 直接建立那样一棵“几乎确定”的树然后再DP是不行的，如下例：
- DFS: 1 2 3 4 5 7 6 8 9 10
- 答案是5.250

- 所以需要O(N^2)解决
- 正解貌似是对每个点各自BFS套DFS
- 没题解，唯一的AC代码也有问题，不会做。

例3 本题还未出

- 看了两道神经病题，换一道简单的放松一下。
- N个数组成数列{a_i}，每次进行以下两种操作之一：
  - 1 l r x：对[l, r]区间内的i，a_i += x
  - 2 l r：求对[l, r]区间内的i，所有a_i的乘积模45045的值
- 怎么做？
- 线段树？
- 块状链表？
- 带修改莫队？
- CDQ整体二分？

例3

- 数据的性质决定结构！
- 45045 = 5 * 7 * 9 * 11 * 13　可以独立取模+CRT
- 以9为例，不妨设一段数列内每个数都加x，那么有：
- 乘积 = (a1+x)(a2+x)…(an+x)　是一个x的多项式
- 若x%3=0，x^2 % 9 = 0，保留最低2次
- 若x%3!=0，x^8 % 9 = 1 = x ^ 0 % 9，循环折叠成8次
- 其实一共是一个(2+8)次多项式！

例3

- 如果是线段树，如何上传?
- 给定两个不同的多项式，两个不同的x，求其乘积后的多项式。
- 似乎是不可能的……

- 块状链表：只需要下传，不需要上传!
- 复杂度：O(N*sqrt(N)*P^2)
- 似乎还是有点高……

例3

- 多项式的本质是什么？已知前N项，求任何一项的一种数列。
- 多项式对于x是循环的，x%9的值才有意义……
- 没有必要维护多项式，维护值！
- 值是可以上传的 => 线段树
- 总复杂度O(N*logN*P)
- 等价关系：区间加很多不同的数，乘积模P值相等

例4 网教“二盖一”

- 给出原序列{ai}和一条极长上升子序列S。
- 求两条不相交的上升子链A和B，使得A和B包括S内的全部元素，且A和B加起来的总长度最长。

- 网络流当然是可以的，但是我们需要更优雅的做法，比如说O(N^2)的？
- 贪心。
- 最天真的想法是取A=S，然后另选一个LIS做B。这当然不对，但是我们可以改进它。

例4

- 其实本来的题是求两条A、B加起来最长，然后想到先求一次LIS再改进，然后挂了……
- 然后就变成了这道题。
- 观察用例我们发现S是分段分布在A和B中的，如果S的一段分布在A（比如说S[l]..S[r]），那么B中就会相应有一段，使其替换下S在A中的那一段后，S依然是一个链（也就是说S[l-1]+这一段+S[r+1]肯定是链）。于是我们考虑动态规划。

例4

- 令DP[i]表示原数列只取前S[i]个且S只取前i个时的答案。不妨假设S[i]在A中，于是我们要枚举j，令j是最大的x使得S[x]在B中，然后DP[i]=DP[j]+???[j][i]。
- 仔细想想，这个写法并不好。
- 这样改一下：
- 令DP[i]表示原数列只取前S[i]个且S只取前i个时的排除S内元素以外的答案。不妨假设S[i]在A中，于是我们要枚举j，令j是最小的x使得S[x]也在A中，DP[i]=DP[j]+F[j-1][i]。S[j]算两遍，但是无所谓。
- F[j][i]表示S[j]和S[i]之间的链，要求取得不在S中的元素个数最多。

例4

- DP是O(N^2)无疑，但是F怎么算?
- 一种算法是不把S从数列中删去，而标记权值w为0。
- 给数列补0和+Inf作为起终元素，然后不止计算S上的元素，而是对全部元素进行区间DP。
- 令G[l][r]表示强制选择a[l]和a[r]作为起终元素的、权值最大的链。
- 枚举最后的选择k，有G[l][r]=max{w[l]+G[k][r]}，其中l<k<=r，a[l]<=a[k]<=a[r]
- 这是O(N^3)的算法。

例4

- 注意到并不是所有的转移都是有意义的……
- 1 2 3 4 5，对G[1][5]而言，不取2而直接取3是毫无价值的。因为取2之后并不影响3的选择。
- 对G[l][r]，只有l之后的、a[k]>=a[l]的、a[k]**严格单调递减**的k序列才是有意义的转移！
- 数学渣表示并不能证明这个复杂度，但是这样转移确实是O(N^2)的区间DP。

- 两次DP过程中记录转移即可得到A和B
- 总复杂度O(N^2)

例5 GCJ的某题

- 一开始你在原点，数轴上有一系列点各自有不同的位置和速度，它们都会想着远离你的方向移动。你的速度一定，每和一个点重合即可抓住之。求抓住所有点的最小用时。

- 这题乍一看似乎非常地不可做，固定点的话我还可以两头ＤＰ，但是点是动的，会破坏我取到的点的位置关系：t1时刻取了距离范围前i的点，t2时刻来说已经取走的未必是距离范围前i的点……
- 真的不能两头ＤＰ？　　性质决定策略！

例5

- 对某一特定的时刻t，假设存在两点P,Q，使得P位置在Q之前，且速度小于Q。那么对t及以后的任何时刻，P位置小于Q。
- 也就是说如果我们在t以后的时间取得Q，那么删去P，解空间不变。
- 对于每一个特定的时刻t，删除不满足要求的点，剩下的点的序列必定满足速度与距离成严格负相关。此时，任何一个点既可以由距离代表，又可以由速度代表，二者各自单调。
- 每一个特定的时刻这个序列是可知的。
- 随时间t递增，序列中的元素单调减少。

例5

- 对于任何一个特定的时刻t，显然有可能最优的策略是走到两个方向之一，取得某一个特定的（序列中的）点，然后回到原点继续决策。
- 在某一特定时刻我们的决策，取得的是在该时刻特定方向上位置小于特定值的全部点，亦即速度大于特定值的全部点。
- 注意速度是不变的，在当前时刻取得的是速度大于特定值v的全部点，在以后的任意时刻我们仍然取得了速度大于v的全部点。
- 对于任意决策任意时刻，我们知道已经取得的点和还未取得的点，它们在速度关系上满足特定阈值关系。

例5

- 这样就可以想到**以速度为状态**的DP了。
- 定义dp[vl][vr]表示负半轴取得速度vl以上点，正半轴取得速度vr以上点并回到原点时，所需要的最小时间。
- 两边分别按速度排序，计算dp。
- 总复杂度O(N^3)
- 以距离为状态进行DP也是可行的。

例6　走迷宫

- 继续换水题放松一下。
- 给一个600x600的地图，要求构造一个有墙的迷宫，使得从S点开始BFS时队列中同时存在的元素个数超过16383
- 纯粹的空白对队列长度要求不高。
- 墙的作用是，通过引导走向，达到更大的一次性爆发。

例6

- 考虑分形算法。
- 一种是二分形。恰好16384。
- 一种是三分形，然后折叠。答案能到17500，但是并不优雅。
- 具体看图即可。

例7 51NOD马拉松4“移数字”

- 开始有一个长度为n的排列，每次可以把第三个数移到排列的最前面（例如把"1 2 3 4 5"变成"3 1 2 4 5"），或者把最后一个数移到排列的最前面（例如把"1 2 3 4 5"变成"5 1 2 3 4"）。目标是把排列从小到大排序。
- 求n的排列中有多少种能够达成目标，多少种不能。结果模任意P，P非质数。
- 排列问题，最常用的不变量是……

例7

- 逆序数的奇偶性！
  - 操作一影响为2 = 0 (mod 2)
  - 操作二影响为(n – 1) = !(n&1) (mod 2)
- N为奇数且逆序对有奇数个时一定无解。
- N为偶数时一定有解。因为我们总可以用操作二把一个数变成3号位然后向前移动2个。2和(n-1)互质，所以我们可以把一个数移动到任意位置。

- 然而，N为奇数且逆序对有偶数个时怎么做?
- 考虑第二个操作实际上使序列成环……

例7

- 归纳法……
- 假设i+1到n可以排序，如果i可以移动到i+1前紧邻，那么i到n可以排序。否则i一定可以移动到i+1后紧邻。
- 如果i>2，那么一定存在小于i的数可以移动到i+1和i之间。亦即无论如何i到n可以排序。
- 当3..n有序时，逆序对只可能是2 1，但是这样逆序对数为奇数。反之逆序对数为偶数时，序列自然有序。
- 也就是说答案是：
  - 偶数：N！
  - 奇数：N！ × 0.5

例7

- 计算阶乘这里有一个技巧。（还记得例1吗？）
- 首先假设N=M^2+X，并令：

$$F(x)=\prod_{i=1}^{M}(x+i)$$

- 然后有： $N!=\prod_{i=0}^{M-1}F(iM)\times\prod_{i=1}^{X}(M^2+i)$
- F(x)可以分治然后用FFT计算。
- 阶乘那一部分是一个多项式多点求值。
- 多点求值的板子从来没见区域赛考过，代码我也没有。不过以防万一给简要说一下原理。

例7

- 假设我们正在处理l..r区间的询问，那么令:

$$G(x) = \prod_{i=l}^{r} (x - q_i)$$

- 于是在这一区间内F(x)和F(x) % G(x)是等效的。
- 用一个分治套上多项式除法搞。除数多项式可以用分治预处理好。
- 总复杂度O(M log^2 M)
- 就是例1写的复杂度。
- 这个算法配合Lucas定理可以处理任意的组合数取模。

例8 2015多校第9场I题

- 给定N个槽位，和其中M个位置{Pi}。
- 你可以采用不同策略将N个槽位劈分成K段，进行若干轮，计算其得分总和作为你的得分。
- 每一轮，机器会随机构造一个N-排列，并将所有{Pi}中的位置进行一次排序，作为实际采用的排列；然后根据你的分段，对每一段分别计算其逆序对数，对全部分段的逆序对数乘积，作为这一轮的得分。
- 你会选取最佳策略，要求求出总得分的期望值。

- (HDU5404 HDU5075)

例8

- 就第一眼感觉来看，和例5相比如何?
- 例5很容易让人联想到简单的两边DP，只是有一个变动的问题在里面，然后感觉非常地不可做……
- 这类题看完题解固然是有一种“果然如此”“其实不难”的感觉，但是现场属于绝对不要碰的类型。
- 为啥？这题杜教没A……

- 再来看例8。本题的操作过程是很复杂的，题目中的“最优化”和“期望”的组合也让人一头雾水。
- 但是我们需要的只是深入的分析而已。表面结构复杂的题，通常没有“卡智商”的要点。何况本题数据范围很小(N为100)。因此这题并非不可做。

例8

- 首先你的策略是毫无意义的。分数是累加的，不是算平均值，当然是越多越好啊。于是我们需要遍历所有的划分，然后求一个总和（如果是算平均，要求最大值）……
- 想到了分段的DP： dp[i][k] -> 把前i个元素分成k段的XXX

- 然后考虑每一段，我们需要知道特定的这一段逆序对个数的期望值……
- 想到了指示器变量：每一对元素构成逆序对的概率相加就是答案。

例8

- 然后考虑段与段的关系。期望非相关可加，而非独立不可乘。
- 考虑[1,2,3][4,5,6]两段，答案是

$$(X_{12}+X_{23}+X_{13})(X_{45}+X_{56}+X_{46})$$
$$=X_{12}X_{45}+X_{12}X_{56}+X_{12}X_{46}+\cdots$$

- 于是乎我们又可以用线性可加的性质，变成每一段取一个数算了。
- 每一段取一个的话，是可以计算/预处理的……
- 关键是，怎么算?

例8

- 我们假设所有的{Pi}都不存在，那么，所有的Xij=0.5
- 所有的{Pi}构成一个固定的大小关系，是一条偏序链
- Xij，ij都不在链上，还是0.5；都在链上，一定是0。
- 一个在一个不在?
- 构成一个有向的树的结构，但是因为方向问题并不好做……
- 加一个容斥 -> 变成单向的边
- 单项树的树根是整棵子树最大或者最小的点。
- 所以每个点有子树大小分之一的概率成立……

例8

- 题解是这么说的不过，主链上的点的排序是固定的啊，是不是不太对啊……
- 当然不对。不过，给答案再乘个M!不就好了。（题外话：杜教题解真心不适合新手看，细节省略太多，新人容易坑。）
- 好的，那么按题解的说法：“如果所有边都连入主链，那么整个图就形成了有根树的结构，这样的图的概率为每个点子树大小乘积的倒数，因为每个点都要在子树里最小。”

例8

- “对于连出主链的边，可以这样转化不连这条边形成的图对应的概率减去将这条边反向之后图对应的概率。因为两个数a,b只可能a<b或者a>b，那么把和将这条边反向之后的概率加起来后，两个点之间就没有限制了。”
- 具体地来说，这段话的意思是，从后面连入主链的边保留，向前面连出主链的边要反向变成从前面连入主链的边减掉，然后再加上完全不连边的对应情况数。
- 需要注意的是任何一段最多连一条边，即使你算的时候算了很多条，那也是多种情况的叠加，边的个数只能+1。

例8

- 具体地说一下go[l,r,k]吧：
  - 0.5的情况是：[l,r]中不在主链上的全部点取C2，乘上概率0.5，边数不加。
  - 从右连入主链的情况是：对主链点i，[i,r]中不在主链上的点的个数，乘以[0,i]主链上点的个数+k（之前连好的边算在i的子树上），边数加一。
  - 向左连出主链的情况是：对主链点i，[l,i]不在主链上的点的个数，乘以[0,i]主链上点的个数+k，取负值给边数加一；然后取[l,i]不在主链上的点的个数，给边数不加（注意这个不乘0.5）
  - 最后我们把每个点为根的子树大小除掉：对边数不加的来说，等于乘上(k+[0,r]中主链点数)阶乘，再除以(k+[0,l-1]中主链点数)阶乘；对边数加一的来说，乘法的阶乘要多加一个1。（为什么？提示：配合前两条多乘的东西考虑）

例8

- 然后是主体的dp[l,j,k]。这个其实没啥好说的，就是枚举每次选取的点（注意每次至少选两个点，不然会乘0），乘go[l,r,k]:不加的转移到dp[r,j+1,k]，加一的转移到dp[r,j+1,k+1]。

- 最后的答案显然是所有的dp[n,m,?]求和，补乘上m的阶乘。

例8

- 至于HDU5075的原题，现在看来已经简单爆了。
- 题目要求是2N个数分成M段，每一段的长度只能是正偶数，然后每一段段内所有偶数位置排序，计算逆序对数乘积作为分数。
- 最后还是求最大化的期望得分。
- 很容易发现，现在两段之间再无任何关系了。

- 考虑每一段（不妨设长度为2n），奇数位的逆序对数期望显然是C(n,2)。偶数位的逆序对数依然是0。

例8

- 对于一奇一偶的情况，不妨设偶数是i，奇数位是j，那么所有的n个偶数和j一共(n+1)个数构成排列，其中j恰好处于某一个特定的位置，概率是$\frac{n!}{(n+1)!}=\frac{1}{n+1}$
- 若j<=i（有i种），那么j在排列中可以处于i之后的(n-i+1)个位置；若j>i（n-i种），那么可以处于i之前的i个位置。取i=1..n求和得到 $\frac{n(n+1)(2n+1)}{6}$
- 上式乘上概率，再加上纯奇数位的结果，得到了转移方案go[n]= (7n - 1) * n / 12
- DP方程：

$$dp[n][k+1]=\sum_{i=1}^{n-1} dp[n-i]\,[k]\times go[i]$$

例8

- 原题N<=2000，10^4组用例，乍看时间似乎是来不及的。
- 但是注意对特定的k，dp[n][k]的生成多项式:

$$DP_k(x)=\sum_{i=k}^{+\infty} dp[i][k]x^i=\left(\sum_{i=k}^{+\infty} go[i]x^i\right)^k$$

- 因此一组答案是可以在O(NlogN)内求出的。（参jcvb课件、Picks博客什么的）
- 即使要求全部解，分治FFT也能O(N^2logN)搞定。

例8

- 没人AC这道题的原因很可能是，模任意质数FFT太烦不想写……
- 300多ms AC的那个人不清楚用的什么方法，应该会更简单。

例⑨ 2015多校第9场G题

- 最后了，来一个简单的吧。
- NxM的矩阵，每个点有一个数。从左上走到右下，每个格子至多经过一次。要求所取数总和最大，求方案。

- 很简单的问题，都做出来了吧?
- 有奇数的时候直接取光所有的点。
- 两个偶数的时候黑白染色，白色点取最小的放过去，把它绕开即可。

总结

- 例1：通过若干种简单的结构，生成全部答案。
- 例2：问题的关键点是“到底哪些点可能造成不同的方案”，抓住这一点仔细分析就出来了。
- 例3：性质决定结构，不是一眼看上去用啥就能用啥的。
- 例4：有效转移点单调的性质是关键。“没有也行”的东西要考虑一下。
- 例5：如上题找出合法链后，还需要发现两个不动点：每个点的速度、链的形状。

总结

- 例6：分形几何学->形状不变性
- 例7：逆序数是不动点
- 例8：分解问题逐步分析，计数中树的形状
- 例9：染色的不动点。

总结

- 常见的拓扑形状(？)：
  - 棋盘或二分图的染色
  - 序列的逆序对
  - 集合中特定元素的大小关系
  - 序列的位置顺序关系
  - 变换不变性（比如说序列的交换，或者环的旋转、翻转）
  - 分形的共形不变性（递归，自相似->放缩法，标度不变型->粗粒平均）