NOIP 中较难的 DP 类问题选讲

150137

2018 年 6 月 19 日

一些前置技能

什么是概率我相信大家都知道，这里只简单的强调一些琐碎的细节：

设某单一随机事件 A 发生的概率为 $P(A)$ ，那么 A 的互斥事件为 A^c，

$$P(A \cup A^c) = 1$$

在事件 A 的前提下发生事件 B 的概率为 $P(B|A)$, 显然有

$$P(B|A) = \frac{P(AB)}{P(A)}$$

对于无交集的事件 A, B，有 $P(A \cup B) = P(A) + P(B)$

对于有交集的事件 A, B,

$P(A \cup B) = P(A) + P(B) - P(A) * P(B)$

一些前置技能

A 事件发生的期望 $E(A)$ 被定义为 $E(A) = P(A) * val(A)$ 即概率乘以取值，期望能够很好的衡量事件的平均程度。
期望具有线性，简单的说即对于两个不相关的事件 A, B,

$$E(A+B) = E(A) + E(B) \tag{1}$$

$$E(AB) = E(A)E(B) \tag{2}$$

考虑到求期望通常需要求出对应的概率，故而大多数情况下倒着 DP 要比正着 DP 更方便，因为对于同一情况，当前概率可选择的情况更少, 更容易求 QAQ。

写在前面

大家处理 *NOIP* 中的期望问题的瓶颈有以下两个

1. 由于陌生，产生畏惧心理
2. 不能够很好的把已有的计数模型进行应用

下面我们通过题目努力攻克这几个问题

BZOJ 1419

桌面上有 A 张红牌与 B 张黑牌，随机打乱放置。现在开始一张一张的翻盘，翻到红牌加一分，黑牌减一分，求在最优策略的情况下的期望得分。

$A, B \leq 5000$

描述手里的牌可以做但是比较麻烦，不妨描述桌子上剩下的牌

描述手里的牌可以做但是比较麻烦，不妨描述桌子上剩下的牌

考虑用 $f(i,j)$ 表示桌面上还剩下 i 张红牌与 j 张黑牌时期望的最佳利益，分类讨论一下

描述手里的牌可以做但是比较麻烦，不妨描述桌子上剩下的牌

考虑用 $f(i,j)$ 表示桌面上还剩下 i 张红牌与 j 张黑牌时期望的最佳利益，分类讨论一下

当 $i=0$ 时 $f(i,j)=0$

当 $j=0$ 时 $f(i,j)=i$

否则的话 $f_{i,j}=max(0,\frac{i}{i+j}(1+f_{i-1,j})+\frac{j}{i+j}(-1+f_{i,j-1}))$

描述手里的牌可以做但是比较麻烦，不妨描述桌子上剩下的牌

考虑用 $f(i,j)$ 表示桌面上还剩下 i 张红牌与 j 张黑牌时期望的最佳利益，分类讨论一下

当 $i=0$ 时 $f(i,j)=0$

当 $j=0$ 时 $f(i,j)=i$

否则的话 $f_{i,j}=max(0,\frac{i}{i+j}(1+f_{i-1,j})+\frac{j}{i+j}(-1+f_{i,j-1}))$

这样的话答案其实就是 $f(n,m)$, 注意到 5000^2 会 MLE，需要滚动一下。

BZOJ 2134

卷子上共用 N 道题目，对于题目 i，共有 a_i 个选项，每个题只能选择一个选项。现在把所有的正确选项都涂到了他的下一道题上，问期望能做对多少道题目

由于一题只会对下一题产生影响且相互独立，可以考虑求出一道有 a 个选项的题写到有 b 个选项的题时的期望得分。

由于一题只会对下一题产生影响且相互独立，可以考虑求出一道有 a 个选项的题写到有 b 个选项的题时的期望得分。

对于相邻两道题共有上面一个题有 a_i 中选择方案，下面一个题有 a_j 中正确的可能，共 $a_i \times a_j$ 中情况作为样本总量

由于一题只会对下一题产生影响且相互独立，可以考虑求出一道有 a 个选项的题写到有 b 个选项的题时的期望得分。
对于相邻两道题共有上面一个题有 a_i 中选择方案，下面一个题有 a_j 中正确的可能，共 $a_i \times a_j$ 中情况作为样本总量
对于做对的情况只有 $\min\{a_i, a_j\}$ 种、
把他们乘在一起再乘上 1 就是答案了。

BZOJ 4318

一共有 n 次操作，第 i 次操作有 a_i 的概率为 1，n 次操作对应为 1 个长度为 n 的 01 串。定义这个 01 串的权值为所有极长的 1 的长度的立方和。求最后得分的期望。

如：010001 权值是 $1^3 + 1^3$,101110 权值是 $1^3 + 3^3$,

首先需要注意，平方的期望不等于期望的平方，因为概率不会随权值平方。

首先需要注意，平方的期望不等于期望的平方，因为概率不会随权值平方。

$$(x+1)^3 - x^3 = 3x^2 + 3x + 1$$

分别求长度的期望，长度平方的期望以及在 i 位置时的期望得分即可。

HDU 4405

大家一起掷骰子玩飞行棋，飞行棋路线长度为 n, 并有 m 条传送门（不相连），每次向前掷出指数个单位。如果正好落在一个传送门上，可以直接穿送到终点。

问走到终点所需要掷的次数的期望。

还是之前的想法，设 $f(i)$ 表示从 i 到终点期望的投掷的次数。

还是之前的想法，设 $f(i)$ 表示从 i 到终点期望的投掷的次数。如果没有路线，那就是

$$E(i) = \frac{\sum_{i=1}^{6} E(i+k)}{6}$$

如果那个位置是一个起点就直接继承终点的次数就可以了。

POJ 2151

现在有 n 支队伍在做题，一共有 m 道题，第 i 个队伍做出第 j 道题的概率为 $p(i, j)$。
现在比赛主办方希望每个队伍都至少做对一道题，且至少有一个队伍做对 k 道及以上的题目，问多大多少概率发生这种情况
$n \leq 30, m, k \leq 1000$

对于每个队伍，$1 - P(\text{爆零})$ 就是不爆零的概率。

对于一个队伍，通过简易的 DP 可以得出其 A 掉 x 道题的概率，那么对于 $x \geq k$ 部分的概率求个和就是超过 k 部分的概率。

直接乘在一起

对于每个队伍，$1 - P(爆零)$ 就是不爆零的概率。
对于一个队伍，通过简易的 DP 可以得出其 A 掉 x 道题的概率，那么对于 $x \geq k$ 部分的概率求个和就是超过 k 部分的概率。
直接乘在一起
然后你就 GG 了，由于这两件事情之间有交集，所以不能直接用概率相乘。

换一个想法考虑，我们设事件 A 为所有队伍都过了至少一道题目，设时间 B 为存在队伍过了 k 道题目由

$$P(AB) + P(AB^c) = P(A)$$

移项就能得到

$$P(AB) = P(A) - P(AB^c)$$

等号左边是所求，右边的部分能通过刚才的 DP 解决。

POJ 3744

给出一条长度为 L 的道路，路上有 n 个地雷.
现在从起点 1 出发，在第 i 个格子时你有 p 的概率走向下一个格子，有 $1-p$ 的概率跨过下一个格子直接走到第 $i+2$ 个格子
求最后不踩雷安全生存下来的概率。
$L \leq 10^8, n \leq 10$

考虑一个比较简单的递推，设安全到第 i 个位置的概率是 $f(i)$，通过简单的推导能够得出：

$$f(i) = p \times f(i-1) + (1-p)f(i-2)$$

这个递推式子收敛，找到一个阈值使得递推到该项之后在误差允许的范围内不再波动。
剩下的部分就把原来的路程分成 $n+1$ 段处理就行了。

POJ 1322

现在你有 C 种颜色的巧克力，你有一大袋巧克力，可以认为每种颜色都是无限多的，
每次操作从袋子中取出一个巧克力放在桌子上，如果桌子上有和其颜色相同的，就把两块颜色相同的巧克力一起吃掉，反之不操作。
问操作 N 次过后桌子上剩 M 块巧克力的概率为多少
$C \le 100, n, m \le 10^6$

$f(m, n)$ 表示选了 n 次之后桌子上剩下 m 块的概率。

找来源略微有点麻烦，考虑算这个东西其他状态的贡献。

$f(m,n)$ 表示选了 n 次之后桌子上剩下 m 块的概率。
找来源略微有点麻烦，考虑算这个东西其他状态的贡献。
$f(m,n+1)=\frac{f(m+1,n)\times(m+1)}{C}+\frac{f(m-1,n)*(C-m+1)}{C}$
当 c 给定的时候这个东西也是收敛的，所以找到阈值之后就可以缩小范围，只保留奇偶性就可以了。

BZOJ 3029

现在摆在你面前的是 n 个关卡，身上有一个容量为 k 的背包，现在你要依次闯关，

已知关卡 i 的挑战成功率为 p_i，如果 i 关卡挑战成功，你可能或获得一块地图碎片或一个一定容量的背包。

你只有挑战成功 L(及以上) 个关卡才能离开这个地方，现在求问你能够带上所有获得的地图碎片离开这个地方的概率是多少 (一块地图碎片的体积为 1)

$k \leq 2000, n \leq 200, L \leq n, p_i \leq 1$

$F[i][j][k]$ 代表现在闯到第 i 关，已经成功闯关了 j 次，现在可用背包容量为 k 的概率

仍然考虑他对别人的贡献。

$F[i][j][k]$ 代表现在闯到第 i 关，已经成功闯关了 j 次，现在可用背包容量为 k 的概率
仍然考虑他对别人的贡献。
如果我们下一次闯关失败，那么有

$$F[next][j][k] = (1 - p[next]) \times F[now][j][k]$$

如果下一次闯关成果，那么有

$$F[next][j+1][k+w[next]] = p[next] \times F[now][j][k]$$

背包最大只需要 n, 所以这个复杂度实际上就是 n^3 的。

JDOJ 2195

现有一只青蛙，初始时在 n 号荷叶上。当它某一时刻在 k 号荷叶上时，下一刻将等概率地随机跳到 1，2，...k 号荷叶上之一，直至跳到 1 号荷叶为止。
求调到第 1 号荷叶期望次数
$n \leq 2 \times 10^3$

$E(i) = \frac{1}{i}\sum_{j=1}^{i} E(j) + 1$
这个式子非常容易拿出来，但是发现会调用 $E(i)$ 本身。

$E(i) = \frac{1}{i}\sum_{j=1}^{i} E(j) + 1$

这个式子非常容易拿出来，但是发现会调用 $E(i)$ 本身。

移项解方程即可。

BZOJ 3566

给出一个树形的电路，共有 n 个节点，节点 i 有 p_i 的概率自动被充电，树边 j 有 q_j 的概率流通电流

求进入充电状态的点的个数的期望

$n \le 5 \times 10^5$

每个点的取值都是 1, 所以如果一个点 i 被点亮的概率是 $f(i)$, 我们的答案其实就是 $\sum_{i=1} f(i)$

每个点的取值都是 1, 所以如果一个点 i 被点亮的概率是 $f(i)$, 我们的答案其实就是 $\sum_{i=1} f(i)$
考虑一个点被充电有几种情况，一种是被自己或者自己儿子充电了，一种是被父亲充了，当然这两件事情是有交集的，最后用有交集的概率公式套一下就好了。

被自己儿子或者自己充电的概率非常好算，设这个概率是 $g(i)$，容易得到

$$g(i) = 1 - (\prod_{i \to j} 1 - g(j)(1 - p_i))$$

就是算没有被任何一个儿子充电的概率，最后用 1 减一下。

第二个部分呢？

我们假设一个点被除了自己和父亲充电的概率是 $h(i)$, 那么容易知道

$$f(fa_i) = g(i) \times q_{fa\to i} + h(i) - h(i)g(i) \times q_{fa\to i}$$

第二个部分呢?

我们假设一个点被除了自己和父亲充电的概率是 $h(i)$, 那么容易知道

$$f(fa_i) = g(i) \times q_{fa \to i} + h(i) - h(i)g(i) \times q_{fa \to i}$$

注意到 $g(i)$ 在刚才我们都已经求完了，考虑第二次从上向下 DP 的时候 $f(fa_i)$ 也已经知道了，剩下的就是解方程求出 $g(i)$ 了。

最后自然有 $f(i) = g(i) + h(i) \times q_{fa_i \to i} - g(i)h(i) \times q_{fa_i \to i}$

n 个人进行 r 轮游戏，每次从左向右轮，第 i 个人有 p_i 的概率被选中，产生 d_i 的贡献，一个人被选过后不会再被选。

求期望的贡献和。

$n \le 250, r \le 132, d_i \le 1000$, 400 组数据。

这个题就比较厉害了，状态的设计比较巧妙。

直接考虑期望并不是一个非常好的方法，考虑求概率然后再求期望。

这个题就比较厉害了，状态的设计比较巧妙。
直接考虑期望并不是一个非常好的方法，考虑求概率然后再求期望。
设 $f(i,j)$ 表示第 i 张牌还有 j 轮机会的概率，考虑第 $i+1$ 张牌，他可能在剩下的 j 轮中被打出，也可能一直都没有被打出去。
如何求在剩下 j 轮中打出去？还是老办法，算一个一直都打不出去的概率，减一下。

$$f(i+1,j) += f(i,j) \times Pow(i+1,j)$$

$$f(i+1,j-1) += f(i,j) \times (1 - Pow(i+1,j))$$

选过的人会被计入答案，算下面式子之后直接更新答案就好

$$ans += f(i,j \times (1 - Pow(i+1,j))) \times d_{i+1}$$

简单的高斯消元

突然偏题【大雾

考虑一个比较简单的情况，给你 n 个未知数，n 个一次方程，求这 n 个未知数，保证有解。

简单的高斯消元

突然偏题【大雾
考虑一个比较简单的情况，给你 n 个未知数，n 个一次方程，求这 n 个未知数，保证有解。
每次用下面的行减去当前行的若干倍，使得第 i 项系数为 0.
到达第 n 行的时候就出一个，最后再反带回去就可以了。

简单的高斯消元

突然偏题【大雾

考虑一个比较简单的情况，给你 n 个未知数，n 个一次方程，求这 n 个未知数，保证有解。

每次用下面的行减去当前行的若干倍，使得第 i 项系数为 0.

到达第 n 行的时候就出一个，最后再反带回去就可以了。

我相信大家一定都会这个方法……这个东西其实就是线性代数的高斯消元的简单形式了。

一些更多的相关线性代数的知识这里就不讲了。

BZOJ 1778

给出一张 n 个点，m 条边的无向图，有一个炸弹从 1 号点出发，每次有 P/Q 的概率爆炸，若不爆炸，那么等概率的向周围的点走一步。

求炸弹在每一个点爆炸的概率。

$n \le 300$

设 $f(i)$ 表示最后在 i 号点爆炸的概率。

设 $f(i)$ 表示最后在 i 号点爆炸的概率。
易知在 i 爆炸的概率为在相邻城市不爆炸的概率乘以从相邻城市转移过来的概率之和，最终乘以爆炸的概率
对于每一个点都有这么一个方程，通过移项之后将会变成一个非常优雅的形式，高斯消元即可。

BZOJ 2337

给出一张带权无向连通图，从 1 号节点出发，每次随机的在相邻边中选择一条走过去，走到 n 停止，路径的权值为路径上所有边权的异或和，问走出路径的期望权值是多少。

看到异或首先想到拆位之后按位确定，然后发现按位确定就只需要知道异或 1 次数的奇偶性。

看到异或首先想到拆位之后按位确定，然后发现按位确定就只需要知道异或 1 次数的奇偶性。

设二进制第 k 位，从 i 到 n 的期望值为 $f[i]$，点 i 的入度为 $cnt[i]$, 那么如果 i,j 间有连边，且当前位的权值为 1，那么 j 对 i 的期望贡献就为 $f[j]/cnt[i]$

如果当前位权值是 0 的话，那么 j 对 i 的期望贡献就为 $(1-f[j])/cnt[i]$

n 个未知数 n 个方程，高斯消元即可。

小结

大体上技巧就是这些，但是更多的问题需要更多的算法相互综合，这里考虑到专题的性质所以就先不讲了。
NOIP 已经把期望概率放到了考点中，大家要攻克难关哦。

什么是数位 *DP*

什么是数位 *DP*

其实就是求满足某些性质的若干进制的数字的种类。

处理这类问题，顾名思义就是按位 *DP*

BZOJ 1024

定义不含前导零且相邻两个数字之差至少为 2 的正整数被称为 windy 数。
求 $[A, B]$ 之间 windy 数的个数。

数位 DP 鼻祖题。显然可以把答案写成两个前缀相减的形式。

数位 DP 鼻祖题。显然可以把答案写成两个前缀相减的形式。先 DP 出所有的方案，设 $f(i,j)$ 代表最高位为 j 的 i 位数中一共有多少个 windy 数，考虑依次枚举最高位，有

$$f(i,j)=\sum_{k=0}^{j-2} f(i-1,k)+\sum_{k=j+2}^{9} f(i-1,k)$$

数位 DP 鼻祖题。显然可以把答案写成两个前缀相减的形式。先 DP 出所有的方案，设 $f(i,j)$ 代表最高位为 j 的 i 位数中一共有多少个 windy 数，考虑依次枚举最高位，有

$$f(i,j)=\sum_{k=0}^{j-2}f(i-1,k)+\sum_{k=j+2}^{9}f(i-1,k)$$

接下来从高到低计算方案，先确定一个最高位，然后找到若干个合法的下一位，在 DP 状态里面直接拿出这个答案。最后对于部分是合法的部分，递归处理即可。

BZOJ 1833

给定两个正整数 a 和 b，求在 $[a, b]$ 中的所有整数中，每个数码各出现了多少次。

$a, b \leq 10^12$

和上一题基本上是一样的，就是多加了一维，公式和上一题基本一样。

和上一题基本上是一样的，就是多加了一维，公式和上一题基本一样。

这个题拿出来主要是告诉大家有的时候有的时候要考虑前导 0 的问题，实现因人而异，但是必须要考虑进去这个问题。

BZOJ 3209

设 $f(i)$ 表示二进制 i 中 1 的个数，求 $f(1)$ $f(n)$ 的乘积。

问题变成了二进制，但是方法完全可以延续下来。

问题变成了二进制，但是方法完全可以延续下来。

考虑如何计算一个长度为 n 的 01 序列的所有情况的 f 乘积即可。

容易知道有 k 个 1 的序列共有 C_n^k 个，所以其实就是若干个快速幂。

剩下的就和前面的一样了。

问题变成了二进制，但是方法完全可以延续下来。

考虑如何计算一个长度为 n 的 01 序列的所有情况的 f 乘积即可。

容易知道有 k 个 1 的序列共有 C_n^k 个，所以其实就是若干个快速幂。

剩下的就和前面的一样了。

这题就是告诉大家其实不一定非得是二进制

BZOJ 4521

规定一个 11 位手机号码是美的当且仅当
1 至少 3 个相邻的相同数字
2 号码中不能同时出现 8 和 4。
3 没有前导零
计算 $[L, R]$ 中所有美的手机号的个数。

数位 dp，从低位到高位，$f(i,j,k,l,m,n)$ 表示第 i 位是 j,$i+1$ 位是 k, 然后把含有 $4,8$ 的状态压进 l，$m(0/1)$ 表示是否已经存在了三个相同的，然后 n 表示当前后缀表示的数字是否小于等于原数字

数位 dp，从低位到高位，$f(i,j,k,l,m,n)$ 表示第 i 位是 j,$i+1$ 位是 k, 然后把含有 $4,8$ 的状态压进 l，$m(0/1)$ 表示是否已经存在了三个相同的，然后 n 表示当前后缀表示的数字是否小于等于原数字
转移的话只需要大力枚举一下新添加进来的一项到底是哪个数字就行了，然后把所有后面的东西都算出来，直接用原状态更新到对应状态上就行了
做法反正是没什么技巧，简单粗暴。

数位 dp，从低位到高位，$f(i,j,k,l,m,n)$ 表示第 i 位是 j,$i+1$ 位是 k, 然后把含有 $4,8$ 的状态压进 l，$m(0/1)$ 表示是否已经存在了三个相同的，然后 n 表示当前后缀表示的数字是否小于等于原数字

转移的话只需要大力枚举一下新添加进来的一项到底是哪个数字就行了，然后把所有后面的东西都算出来，直接用原状态更新到对应状态上就行了

做法反正是没什么技巧，简单粗暴。

这个是给大家一个启示，如果有很多个限制可以考虑用一个多维的状态来维护。

lucas 定理与数位 *DP*

相信大家一定都会 lucas 定理的递归形式，不过大家也应该考虑到这个东西其实就是把 n, m 都写成 p 进制然后对应每一位做组合数最后乘在一起。

BZOJ 4737

给定 n, m, k, 问对于所有的 $0 \le i \le n, 0 \le j \le \min\{i, m\}$ 的 C_i^j 有多少是 k 的倍数

$n, m \le 10^18$, k 是质数，答案对 $10^9 + 7$ 取模。

如果刚才的都会了这个就是裸题了。

如果刚才的都会了这个就是裸题了。
考虑写成 k 进制后，合法的需求就是下面的数字的每一个都比上面的大于或等于。
对于这题 $f[i][0/1][0/1]$ 表示当前已决策高于第 i 位的部分，已决策部分是否与 n, m 相等，满足条件的方案数。
这个题的难度还是远超于 *NOIP* 的，大家了解就好了。

状压 DP

什么是状压 DP

就是一个东西他现有的状态非常少，我们可以进行压缩后放在状态里进行动态规划。

状压 DP

什么是状压 DP

就是一个东西他现有的状态非常少，我们可以进行压缩后放在状态里进行动态规划。

常见的就是进制状压和多维状压。

进制状压是把一个状态用一个其进制数来表示，而多维状压则是每一维 lmt 不同但是乘起来非常少，所以若干维来描述这个状态。

总的来说其实状压本质就是一个枚举所有有效状态的大暴力

邦德 I

有 n 个人和 n 个任务，对于人物 i 与任务 j，其完成成功的概率为 $p_{(i,j)}$，现在请你安排完成任务方案，使得所有任务都被成功完成的概率最大。

$n \leq 20$

设 $f(i, sta)$ 表示前 i 个人分配完任务的状态为 sta 的最大概率。sta 写成二进制后每一位表示一个任务，如果这个任务被分配了就是 1，否则就是 0

设 $f(i, sta)$ 表示前 i 个人分配完任务的状态为 sta 的最大概率。
sta 写成二进制后每一位表示一个任务，如果这个任务被分配了就是 1，否则就是 0
更新很简单，找到 sta 里面没完成的一个任务，用
$f(i, sta) + p_{(i,j)}$ 更新就好了。
注意到转移的前提是状态合法，即 sta 二进制中 1 的个数等于 i.

设 $f(i, sta)$ 表示前 i 个人分配完任务的状态为 sta 的最大概率。
sta 写成二进制后每一位表示一个任务，如果这个任务被分配了就是 1，否则就是 0
更新很简单，找到 sta 里面没完成的一个任务，用 $f(i, sta) + p_{(i,j)}$ 更新就好了。
注意到转移的前提是状态合法，即 sta 二进制中 1 的个数等于 i.
我相信大家都会线性求 $[1, n]$ 中每个数字二进制中 1 的个数

Grape

一个农场的葡萄架上挂着 n 串葡萄，若取一个葡萄就会获得与其相应的美味值。对于连续的 k 串葡萄，最多取 b 串，最少取 a 串，

问能够获得的最大美味值为多少

$n \leq 100000, a \leq b \leq k \leq 10$

显然是压一下前面 k 个葡萄的情况。
从前一个状态到后一个状态只需要把原状态左移一位与上一个全 1 的规定长度的数字。

显然是压一下前面 k 个葡萄的情况。
从前一个状态到后一个状态只需要把原状态左移一位与上一个全 1 的规定长度的数字。
转移方程显然的不能再显然了……
注意这次的合法性要考虑其二进制 1 的个数在 $[a, b]$ 之间。

BZOJ 1072

给一个数字串 s 和正整数 d, 统计 s 有多少种不同的排列能被 d 整除（可以有前导 0）。

$\|s\| \le 10, d \le 1000$

s 这么小肯定是状压一下 s. 然后还和余数相关所以在第二维加上余数进行统计。

$f(sta, p)$ 代表当前已选数字的状态为 sta，模以 d 后余数为 p 时共有多少种排列方式。

考虑每次在前面添加一个数字，$p = (p + 10^{pos})\% modsta$ 就是多了一个 1. 新状态加等于现在这个状态的方案数。

s 这么小肯定是状压一下 s. 然后还和余数相关所以在第二维加上余数进行统计。

$f(sta, p)$ 代表当前已选数字的状态为 sta，模以 d 后余数为 p 时共有多少种排列方式。

考虑每次在前面添加一个数字，$p = (p + 10^{pos})\% mod sta$ 就是多了一个 1. 新状态加等于现在这个状态的方案数。

一个比较需要注意的事情是需要去重，因为这么统计的时候 121 这样的数字两个 1 位置不同，被计算了两次，但是实际上答案应该只计一次。

s 这么小肯定是状压一下 s. 然后还和余数相关所以在第二维加上余数进行统计。

$f(sta, p)$ 代表当前已选数字的状态为 sta，模以 d 后余数为 p 时共有多少种排列方式。

考虑每次在前面添加一个数字，$p = (p + 10^{pos})\% modsta$ 就是多了一个 1. 新状态加等于现在这个状态的方案数。

一个比较需要注意的事情是需要去重，因为这么统计的时候 121 这样的数字两个 1 位置不同，被计算了两次，但是实际上答案应该只计一次。

去重只需要用总的方案数除以每个元素出现次数的阶乘。证明考虑标号统计一个元素的不同种类，以及元素直接互不影响即可。

s 这么小肯定是状压一下 s. 然后还和余数相关所以在第二维加上余数进行统计。

$f(sta, p)$ 代表当前已选数字的状态为 sta，模以 d 后余数为 p 时共有多少种排列方式。

考虑每次在前面添加一个数字，$p = (p + 10^{pos})\%modsta$ 就是多了一个 1. 新状态加等于现在这个状态的方案数。

一个比较需要注意的事情是需要去重，因为这么统计的时候 121 这样的数字两个 1 位置不同，被计算了两次，但是实际上答案应该只计一次。

去重只需要用总的方案数除以每个元素出现次数的阶乘。证明考虑标号统计一个元素的不同种类，以及元素直接互不影响即可。

BZOJ 1097

给出一张 n 个点 m 条边的无向连通图，现在一个人从 1 号点出发，想要遍历 2 $k+1$ 号节点（不一定要按照顺序）然后到 n 号节点去，但是有若干个对于这 k 个点游历顺序的要求，顺序要求的格式为 $(D1, D2)$，代表在到达 $D2$ 之前必须到达 $D1$

求最短距离。

先以前 $k+1$ 个点为原点跑最短路。考虑 $f(sta,j)$ 表示已经去过的前 k 个点的状态是 sta 并且停留在 j 位置的最短距离。
对于那些限制，我们先提前搞出每个点的限制的总的集合为 lmt, 每次看看是不是对应的都有 1 了就好。

先以前 $k+1$ 个点为原点跑最短路。考虑 $f(sta,j)$ 表示已经去过的前 k 个点的状态是 sta 并且停留在 j 位置的最短距离。
对于那些限制，我们先提前搞出每个点的限制的总的集合为 lmt, 每次看看是不是对应的都有 1 了就好。
这个 DP 顺序混乱，可以直接记忆化搜索。

枚举子集

考虑枚举 $[1,n]$ 每个数字的二进制下的子集。假设 n 二进制长度为 l.

直接枚举是 4^l 或者说是 n^2

枚举子集

考虑枚举 $[1, n]$ 每个数字的二进制下的子集。假设 n 二进制长度为 l.
直接枚举是 4^l 或者说是 n^2
事实上存在 3^l 的枚举方法，不过 *NOIP* 并不需要，当然这个东西其实很短。

一个小结

考虑到大家可能刚开始接触状压所以里面并没有什么太深的变形。事实上更难的状压题也往往会和一些其他的什么算法或者想法相结合。

一个比较简单的例子是如果转移的方式是一定的即一个状态 sta_1 一定会转移到下一个状态 sta_2, 那么我们可以用矩阵乘法来优化这个过程。

另外一个角度是有一些状态是没有什么用处或者干脆是不可能被用到的，转移的时候可以先预处理出所有可行的状态来优化复杂度。

BZOJ 4197

给定 n 个数，规定一组数的价值为这组数的最小公倍数，现在求所有组合的价值的和。这样的组合一共有 2^n-1 种，$n \le 500$，每个数字都 ≤ 200

对于任意一个 ≤ 200 的数字，> 13 的质因子至多只有一个，≥ 13 的约数的个数最多有多少个是可以计算的。

$f(i, j, k, l, m, n)$ 表示 2 用了 i 个，3 用了 j 个，···,13 用了 n 个的方案数.

对于任意一个 ≤ 200 的数字，> 13 的质因子至多只有一个，≥ 13 的约数的个数最多有多少个是可以计算的。

$f(i,j,k,l,m,n)$ 表示 2 用了 i 个，3 用了 j 个,⋯,13 用了 n 个的方案数.

我们只需要把每个数字质因数分解，然后每一维都取一下最大值方案直接加一下就行了，对于大于 13 的质因子，我们可以直接加上这个质因子大小的方案数。

然后是否含有 > 13 的质因子就大力再开一维就行了

区间 DP

啥是区间 DP, 区间 DP 就是在一个区间上进行 DP, 通常 DP 的定义都类似为 $f(l, r)$ 表示在 $[l, r]$ 这段区间中某最优化子结构的最优取值是多少。
当然有的时候 DP 也可以用记忆化搜索来实现，便于理解。
我们看几个例题大家感受一下。

矩阵乘法

给定一行 n 个矩阵，给他们添加括号，求他们进行乘法最后变成一个矩阵的最小次数。

我们认为两个矩阵 $A(m \times n) \times B(n \times p)$ 的乘法次数为 $m*n*p$ 次。

数据保证任意两个相邻的矩阵都是可乘的。

帮助大家理解的题，并不是很难。

设 $f(l,r)$ 表示把 $[l,r]$ 区间矩阵都乘完的价值。

帮助大家理解的题，并不是很难。

设 $f(l, r)$ 表示把 $[l, r]$ 区间矩阵都乘完的价值。

$$f(i, j) = \min\{f(i, k) + f(k, j) + a_i * a_j * a_k\}$$

BZOJ 1742

给定一个数轴，初始在位置 p, 移动一个单位花费时间 1，有 n 坨草 ($n \leq 3000$)，约瑟芬需要吃掉所有的草
定义一坨草的腐败值为吃掉他的时刻，求最小腐败值之和，吃草不需要时间

容易发现一头牛如果左边最左的草吃的是 l, 右面最右的草是 r, 那么 $[l, r]$ 之间的草一定都被吃完了，因为路过的时候就可以顺便吃掉了。

容易发现一头牛如果左边最左的草吃的是 l, 右面最右的草是 r, 那么 $[l, r]$ 之间的草一定都被吃完了，因为路过的时候就可以顺便吃掉了。

$f(i, j, 0/1)$ 表示牛吃完了 $[l, r]$ 区间的草，最后停在左面或右面的最小腐败值。

容易发现一头牛如果左边最左的草吃的是 l, 右面最右的草是 r, 那么 $[l, r]$ 之间的草一定都被吃完了，因为路过的时候就可以顺便吃掉了。

$f(i, j, 0/1)$ 表示牛吃完了 $[l, r]$ 区间的草，最后停在左面或右面的最小腐败值。

分 4 种情况讨论即可。

BZOJ 1055

给定一堆（< 64）映射，每个映射为 $WING$ 中的一个字母映射为 $WING$ 中的两个字母。
现在给定一个字符串 (长度 < 200)，问最开始可能是从哪个字母变形过来的。
如果有多种情况就都输出一下。

bool 型区间 DP, 考虑函数 $able(l, r, s)$ 表示 $[l, r]$ 是否能缩成 s 串。
所以对于一段区间 $[l, r]$ 枚举中间点 k, 以及最终可能组成的组合，然后记忆化搜索下去就可以了。

bool 型区间 DP, 考虑函数 $able(l, r, s)$ 表示 $[l, r]$ 是否能缩成 s 串。
所以对于一段区间 $[l, r]$ 枚举中间点 k, 以及最终可能组成的组合，然后记忆化搜索下去就可以了。
时间复杂度是 $64 * 200^3$, 还是跑的出来的。

给定一个压缩法则，$aaaaa$ 可以写成 $5(a)$
现在给你一个串，对于一个类似的可以压缩的串，你可以选择压缩或者不压缩，最终要使压缩后的字符串的长度尽可能小。循环节长度可以自己定义。如 $AAAAAAAAABABABCCD$ 的最小压缩就是 $9(A)3(AB)CCD$, CC 压缩后占得长度更大所以选择不压缩。

$f(l,r)$ 表示 $[l,r]$ 区间压缩后的最小长度。

$f(l,r)$ 表示 $[l,r]$ 区间压缩后的最小长度。
在不考虑缩写的情况下显然有

$$f(l,r)=\min\{r-l+1, f(l,k)+f(k+1,r)\}$$

$f(l, r)$ 表示 $[l, r]$ 区间压缩后的最小长度。
在不考虑缩写的情况下显然有

$$f(l, r) = \min\{r - l + 1, f(l, k) + f(k + 1, r)\}$$

但是考虑这个可能不是所有的情况，还要考虑 $(k + 1, r)$, 可以由 (l, k) 重复而来，于是在验证可以匹配后，加上 $f(l, r) = \min\{f(l, k) + 2 + calc((r - l + 1)/(k - l + 1))\}$ $calc$ 是计算一个数字十进制的位数。

小结

事实上类似的题目还有很多，不过大都大同小异。

区间 *DP* 存在一些单调决策优化（四边形不等式），但是 *NOIP* 不考察，所以先略去。

更多的题目希望大家能掌握套路后在实践中自己练习。