动态规划选讲 2024.7

动态规划的状态设计

序列一维 dp ，背包，区间 dp，树上 dp，DAG dp，状压 dp，数位 dp

动态规划求解过程的优化

矩阵乘法，高斯消元，单调栈/单调队列，斜率优化，决策单调性，wqs 二分，线段树/树状数组/st表/平衡树，倍增

有技巧的状态设计

费用提前计算，断环为链，交换结果和状态维度，容斥原理

基于题目性质的状态设计

[AHOI2009] 中国象棋

n 行 m 列的棋盘上放若干个互不攻击的炮的方案数。$n,m \le 100$

设 $f_{i,j,k}$ 表示前 i 行决策完毕，有 j 列没有炮，k 列有一个炮，$m-j-k$ 列有两个炮的方案数。枚举下一行摆放炮的数量和位置即可。组合数描述方案数。时间复杂度 $O(nm^2)$ 。

code : from_yixiuge777

[NOIP2021] 数列

给定整数 n,m,k，和一个长度为 $m+1$ 的正整数数组 $v_0,v_1,\ldots,v_m$。对于一个长度为 n，下标从 1 开始且每个元素均不超过 m 的非负整数序列 $\{a_i\}$，我们定义它的权值为 $v_{a_1}\times v_{a_2}\times\cdots\times v_{a_n}$。当这样的序列 $\{a_i\}$ 满足整数 $S=2^{a_1}+2^{a_2}+\cdots+2^{a_n}$ 的二进制表示中 1 的个数不超过 k 时，我们认为 $\{a_i\}$ 是一个合法序列。计算所有合法序列 $\{a_i\}$ 的权值和对 998244353 取模的结果。

设 $f_{i,j,k,l}$ 表示要填入序列的第 i 个数，这时应该填入的大小为 j , 即填入后 $a_i=j$ ，不加上这里已经有 k 个位置上二进制位是1，这一位从下面进上来累计有 l 个1在这一位上。这时的所有序列当前的总价值。

枚举要填多少位这一个大小在这里。设要一块填 v 位，都放上 j 数字。如果这一个数字略过不放，转移有 $fi,j,k,l->f_{i,j+1,k+0/1,l/2}$ 。如果填，转移有 $f_{i,j,k,l}->f_{i+v,j+1,k+0/1,(l+v)/2}$ 。

注意有转移的系数，因为填上了 v 个数字，序列的价值要 $*(V_j)^v$ 。而且序列的种类数变多了，放上这一位序列的种数 $*C_{i+v-1}^{v}$ 。具体的系数可以这么想：设原来 $f_{i,j,k,l}$ 背后有 g 种情况，每一种情况的价值为 f' ， $\sum f'=f$ 。现在相当于加上了 $\sum f'*(V_j)^v*C_{i+v-1}^{v}$ ， 也就是$f_{i,j,k,l}*(V_j)^v*C_{i+v-1}^{v}$ ，发现与 g 没有关系，并不需要保留下来。转移的系数就是 $(V_j)^v*C_{i+v-1}^{v}$ 。

最后统计答案的时候，有些位还在 l 里保留着，没有进上去，实际上所有的二进制位多少为 $k+popcount(l)$ 。也就是说，最终答案 $ans=\sum f_{n+1,j,k,l}$ ， $k+popcount(l)\le K$ 。

预处理出 $(V_j)^v$ 和 $popcount$ ，状态总数为 $i*j*k*l=n*m*n*n=n^3m$ ，转移时间为 $v=n$ ，总时间复杂度为 $O(n^4m)$ 。

code : from_yixiuge777

[NOIP2021] 方差

单调不降正整数序列，每次操作将 a_i 变成 $a_{i-1}+a_{i+1}-a_i$。若干次操作之后序列方差最小值是多少。 $n\le 10000, a_i\le 600$。

题目的操作可以转换为交换差分数组。想让方差最小，就要最集中，于是差分数组除1之外单峰，先单减再单增。从中间开始由小到大填入差分数组，设 $f_{i,j}$ 表示填了前 i 个数，现在已经填的数的和是 j ，平方和最小的值。转移的时候枚举这一个差分数放到了左边还是右边，如果放到右边，相当于放上了一个数 s_i ，如果放到左边，相当于让所有数加上了 b_i ，利用和也可以快速处理出信息。具体的方程：

$$f_{i,j}=f_{i-1,j-s_i}+s_i^2$$

$$f_{i,j}=f_{i-1,j-i*b_i}+i*b_i^2+2*b_i*(j-i*b_i)$$

上面的方程是放到右边，下面的是放到左边。b_1 最后放，只能放到左边。转移即可，当 $b_i=0$ 时一开始的一段都完全没有转移，因此有用的 b_i 只有 $\min(n,a_i)$ 个。时间复杂度 $O(\min(n,a)na)$。

code : from_yixiuge777

树上 dp：

[JSOI2018] 潜入行动

树。在一个点放监听设备可以覆盖与其直接连边的相邻的点，但不会覆盖自己。总共放 k 个设备，覆盖所有节点的方案数。 $n\le 100000, k\le 100$

注意有第二维限制的树上背包是 $O(nk)$ 的。证明：合并两个子树可以看成是前一个子树 dfs 序较大的 k 个和后一个子树 dfs 序较小的 k 个合并，这样时间复杂度和对数相等。对于每个点，只会和左右相邻的 $2k$ 个点配对。时间复杂度 $O(nk)$。

[PKUWC2018]随机游走

有根树。每次询问树上随机游走经过一个点集中所有点至少一次的期望步数。 $n\le 18$。

相当于求一个点集中经过最后一个点的期望步数。比较困难，使用 min-max 容斥转化为经过点集中第一个点的期望步数。枚举点集 S 后进行树上 dp ：设 f_i 表示从 i 开始随机游走经过 S 中第一个点的期

望步数。如果 $i \in S$ ，则 $f_i = 0$ 。转移时枚举走到儿子或者走到父亲。这里转移只能使用高斯消元解决，时间复杂度 $O(2^n n^3)$ 。

对于树上随机游走的高斯消元求解，可以通过以下手段做到线性：儿子的 dp 值是父亲的 dp 值的一次函数。具体推导一下，设 $f_i = A_i \times f_u + B_i$ ，其中 u 是 i 的父亲。

$$f_i = \frac{1}{deg_i}(f_u + \sum_{v \in son_i} f_v) + 1$$

$$f_i = \frac{1}{deg_i}(f_u + (\sum_{v \in son_i} A_v) \times f_i + \sum_{v \in son_i} B_v) + 1$$

设 $sumA_v = (\sum_{v \in son_i} A_v), sumB_v = (\sum_{v \in son_i} B_v)$，并看做常数，上述式子中只存在 f_u 和 f_i 作为变量。整理得到

$$f_i = \frac{1}{deg_i - sumA_v} \times f_u + \frac{deg_i + sumB_v}{deg_i - sumA_v}$$

由此可以直接树形 dp 求解。时间复杂度 $O(n2^n)$ 。最后 min-max 容斥可以做一遍高维前缀和处理出所有答案。

为什么说求点集中经过的最后一个点的期望比较困难呢？其实也能求。设 $f_{S,i}$ 表示从 i 出发经过 S 中所有点的期望步数。那么转移的时候枚举走出去的方向可能会让 S 变小。从小到大枚举 S 求解，如果 S 变小便将这部分贡献视为常数，S 不变便进行树上高斯消元，套用上述过程做到 $O(n2^n)$ 。所以 min-max 容斥不是必须的，至少本题没有体现出他的必要性。

code : from_yixiuge777

[PKUWC2018]Minimax

树，每个点最多有两个儿子。叶子结点有一个权值，每个叶子节点的权值互不相同。内部节点如果只有一个儿子，其继承其儿子的权值；如果有两个儿子，有 p_i 的概率取两个儿子中较大的权值，$1 - p_i$ 的概率取两个儿子中较小的权值。设根可能的权值中第 i 小的权值为 v_i ，概率为 D_i ，求 $\sum_i i \times v_i \times D_i^2$ 。$n \le 300000$ 。

设 $f_{i,j}$ 表示 i 号节点取到 j 数字的概率。设 m 为叶子节点的个数，ls 表示左儿子，rs 表示右儿子，则

$$f_{i,j} = f_{ls,j} \times (p_i \sum_{k=1}^{j-1} f_{rs,k} + (1-p_i) \sum_{k=j+1}^{m} f_{rs,k}) + f_{rs,j} \times (p_i \sum_{k=1}^{j-1} f_{ls,k} + (1-p_i) \sum_{k=j+1}^{m} f_{ls,k})$$

发现转移需要用到的系数是一个前缀和形式。线段树合并。从左往右一遍合并一遍维护前缀和。维护乘法标记。时间复杂度 $O(n \log n)$ 。

code : from_yixiuge777

静态顶树(toptree) sd2023一轮省集day4t2

树，点有点权。一个连通块的价值是点权值的最小值，对于一个边集，价值是所有连通块价值之和。求所有边集的价值之和。 $n \le 300000$

设 $f_{i,j}$ 表示 i 的子树中，与 i 相连的连通块的最小值是 a_j 。这个也可以线段树合并优化。

code : from_yixiuge777

[APIO2016] 烟火表演

树上边有边权。更改边权代价是更改后相差的绝对值。边权修改后非负。让所有叶子的带权深度相同的最小修改代价。 $n \le 300000$ 。

设 $f_{i,x}$ 表示 i 子树叶子节点深度均为 x 的修改代价。每次相当于 (min,+) 卷积上一个绝对值函数，归纳知这是一个凸函数。观察看让 $f_{i,x}$ 加上一条到父亲的边去贡献后会对函数值造成什么影响。

设 $[L,R]$ 为函数中水平斜率为 0 的一段。加入边长长度为 m 。下面直接用 $f(x)$ 代表 $f_{i,x}$ 处 dp 值。

- $x \in [L+m, R+m]$ ，此时边长不需改动，函数值仍为最小值 $f(x)$。
- $x > R+m$ ， 当边长增加 1，函数改变量大于等于 1 。于是我们修改边长使得 $f(R)$ 进行贡献。相当于将原函数 $x > R$ 的部分斜率修改为 1 ，然后向右平移 m 个单位。
- $x \in [L, L+m)$ ，和上一种情况类似。当边长减少 1 ，函数改变量大于等于 1 。于是修改边长让 $f(L)$ 进行贡献。相当于在 $[L, L+m)$ 部分插入了一条斜率为 -1 的直线。
- $x < L$ 直接将边长修改成 0 。相当于将原函数图像上移 m 个单位。

由此完成了变换。叶子处的函数图像是一个绝对值函数，可以看做中间斜率为 0 的段存在，斜率单增，每段相差 1 。而变换之后斜率依然每段相差 1 ，并且拐点数量最多加一。于是我们只维护斜率加一的拐点，如果这一斜率没有对应，就加入两个相同的拐点。合并两个函数即让其拐点集合合并。现在看看上面加入一条边的操作让函数的拐点集合有什么变化：

- 让后面一段上升的斜率为 1 。

因为从儿子合并上来的最后一段斜率均为 1 ，所以删掉 $|son|-1$ 个最大的拐点即可。

- 斜率为 0 的一段平移 m 个单位。

做完上一个操作之后最大的两个拐点就是斜率为 0 的两个端点了。将其加上 m 即可。同时也顺带完成了中间插入一段斜率为 -1 的直线的任务。

需要维护弹出最大值，合并两个集合。可并堆实现。最终 $f(0)$ 即为所有边权之和，容易求出函数最小值。时间复杂度 $O(n\log n)$ 。这种维护函数拐点的方法被称为 Slope Trick。

code : from_yixiuge777

[八省联考 2018] 林克卡特树

树，边有边权。在树上寻找 $k+1$ 条不交链，和最大多少。$n,k\le 300000$ 。

设 $f_{i,j,0/1/2}$ 表示 i 子树内选出了 j 条完整的链，其中 i 的度数为 0/1/2 ，这时的最大价值。很难优化掉 j 这一维度。发现这是个凸函数，wqs 二分。时间复杂度 $O(n\log w)$。

code : from_yixiuge777

划分序列（sequence）sd三轮省集day6t1

一个区间的价值是 $[OR]xor[AND]$。划分序列成 k 段使得权值和最大。 $n,k\le 100000,a_i\le 2^{30}$ 。

设 $f_{i,j}$ 表示为前 i 个数分成 j 段的最大价值。位运算有关，区间价值只会变化 $O(\log a)$ 次，可以优化到 $O(nk\log a)$。

$f_{n,k}$ 关于 k 是凸的。wqs 二分优化到 $O(n\log k\log a)$。

code : from_yixiuge777

[IOI2014] holiday 假期

有 n 个城市，第 i 个城市有 a_i 个景点。从 st 出发，经历 d 天假期，每天可以选择向左或向右走一个城市，或是参观这个城市的所有景点。一个城市的景点只能参观一次。求参观景点数目最多是多少。$n\le 100000$ 。

走的路线一定是先往左走一段，然后掉头往右走。反过来同理。枚举走到的左端点，对应的右端点一定不降。因为走到的右端点一定是要参观的，而区间大小变小只会增加决策，不会减少决策。主席树维护查询区间前 k 大值，分治求解决策单调性 dp 。时间复杂度 $O(nlog^2n)$ 。

code : from_yixiuge777

[国家集训队] Crash 的文明世界

树。对于每个点 x 求 $Ans_x=\sum_{i=1}^n dis(i,x)^k$ 。 $n\le 50000$， $k\le 150$ 。

暴力做，设 $f_{x,j}=\sum_{i\in subtree_x} dis(i,x)^j$ ，从儿子方向转移过来要处理 $\sum_i(dis(i,x)+1)^j$ 的式子。二项式展开，变成 $\sum_i\sum_{r=0}^j\binom{j}{r}dis(i,x)^r$ 。交换求和符号得到 $\sum_{r=0}^j\binom{r}{j}f_{x,r}$ ，可以求解。时

间复杂度 $O(nk^2)$ 。计算其他点的答案做换根 dp。

麻烦在于普通幂不容易递推，二项式展开后计算多一个关于次数的复杂度。动态规划结果记下降幂，斯特林反演得到普通幂答案。具体的

$$Ans_x = \sum_i dis(x,i)^k = \sum_i \sum_{j=0}^{k} \begin{Bmatrix} k \\ j \end{Bmatrix} \binom{dis(i,x)}{j} j! = \sum_{j=0}^{k} \begin{Bmatrix} k \\ j \end{Bmatrix} j! \sum_i \binom{dis(x,i)}{j}$$

$$f_{x,j} = \sum_{i \in subtree_x} \binom{dis(x,i)}{j}$$

$$f_{x,j} = \sum_{son} \sum_{i \in subtree_{son}} \binom{dis(i,son)+1}{j} = \sum_{son} f_{son,j} + f_{son,j-1}$$

时间复杂度 $O(nk)$ 。计算其他点的答案使用换根 dp 。

code : from_yixiuge777

状压:

「C.E.L.U-01」门禁

无向完全图上每条边有出现概率。求连通块个数期望。

设 f_S 表示 S 中的点构成一个连通块的概率。正难则反，枚举集合中和一个固定点联通的一个连通块，使其和集合其他部分均无边相连。暴力是 $O(3^n n^2)$ ，瓶颈在于每次给出两个集合，求他们两两无边的概率。可以折半：将点分成两半，处理出两两之间集合间的答案，询问的时候四块合起来即可。时间复杂度 $O(3^n + 2^n n^2)$ 。

code : from_yixiuge777

[省选联考 2021 A/B 卷] 滚榜

n 支队伍滚榜。第 i 支队伍封榜前通过了 a_i 道题目。排行榜按照 a_i 从大到小排名，通过题目数量相同则编号小的队伍排名靠前。现按照封榜后通过题目数 b_i 不降的顺序宣布每个队伍最终过题数 $a_i + b_i$ 并实时更新排行榜。每次公布后，该被公布结果的队伍都成了排行榜上第一名。已知封榜后所有队伍总共通过 $\sum_i b_i = m$ 题，最终排行榜上的队伍排名情况可能有多少种？ $n \le 13, m \le 500$

$f_{S,i,j,k}$ 表示 S 集合内的队伍已经宣布了结果，目前排行榜第一（最后一个揭榜队伍）是 i ，通过了 j 题，剩余队伍总共通过 k 题，排行榜上的排名情况数量。转移时枚举下一个队伍及其通过题目数量。时间复杂度 $O(2^n n^2 m^3)$ 。这样不仅时间爆炸正确性也不对。每种排名可能有多种 b_i 方法导致，而我们实际上计数的是 b_i 排列，二者如不能做成一一映射则答案错误。

我们只需要求解最终排行榜上排名情况数量，不需要在乎 b_i 的排布方式。则我们尽可能让 $\sum b_i$ 小。此时队伍的排列方案能计算出唯一的最小的 b_i ，下一个队伍通过题目数量可以被计算得出。时间复杂度 $O(2^n n^2 m^2)$ 。

如果一个队伍通过了 b_i 道题目，则后续队伍至少通过 b_i 道题目。如果将后续队伍的这些题目费用提前计算，便不需要记录第一名通过题目数量。下一个队伍多通过题目数量也可以通过计算得出。j 维度被优化掉了。时间复杂度 $O(2^n n^2 m)$ 。

code : from_yixiuge777

为什么一定要 dp 呢？meet-in-the-middle，假设前一半的最后一个 b 是 x ，则后面一半的 b 也要加上 x 。这样 m 甚至可以出的更大！我没写代码，在题解区翻到了 @gyh20 的题解刚学会的。

【清华集训2014】主旋律

有向图，多少种边集使得其为强连通图。$n \le 15$ 。

设 f_S 为 S 中的点连成强连通的方案数。 规定 $E[S1, S2]$ 表示入度在 $S1$ 里，出度在 $S2$ 里的边的数量。对于任意一种连边方式，将其缩点之后形成一个 DAG ，有一些出度为 0 的 scc。很难直接枚举这些 scc 集合，因为不好控制其他的点不形成出度为 0 的 scc 。二项式反演，钦定一部分 scc 集合。设钦定了 T 中的点作为出度为 0 的 scc 集合，则

$$2^{E[S,S]} = \sum_{T \subset S, T \ne null} g_T * 2^{E[S-T,S-T]} * 2^{E[S-T,T]}$$

$$g_T = \sum_{s_1 \cup s_2 \cup \ldots \cup s_k = T} coef_k \times \prod f_{s_i}$$

这里的 $coef_k$ 为容斥系数，与 k 有关。对于一种有 k 个出度为 0 的 scc 的方案，只需要被统计一次，则

$$\sum_{j=1}^{k} \binom{k}{j} coef_j = 1$$

二项式反演得到 $coef_k = (-1)^{k+1}$ 。看出 $g_{null} = -1$ 。现在可以开始求解了！注意第一个式子没有 f ，可以直接求出 g 。

$$g_S = 2^{E[S,S]} - \sum_{T \subsetneq S, T \ne null} g_T * 2^{E[S-T,S-T]} * 2^{E[S-T,T]}$$

将容斥系数代入观察 f 和 g 的关系：

$$g_T=\sum_{s_1,s_2,\ldots,s_k}(-1)^{k+1}\times\prod f_{s_i}$$

枚举第一个元素所在的集合

$$g_T=(-1)*\sum_{s_1\subset T,s_1\neq null}f_{s_1}g_{T-s_1}$$

提取出 f_T

$$f_T=g_T+\sum_{s_1\subsetneq T,s_1\neq null}f_{s_1}g_{T-s_1}$$

以上。时间复杂度 $O(3^n)$。

code : from_yixiuge777

sol : from_this

[NOI2015] 寿司晚宴

2 到 n 的数字，两个人各自选择一个集合，两个集合所有数字对应互质的方案数。 $n\leq 500$

因为 $23^2=529>500$ ，所以 23 及以上的质数在一个数字中最多出现一次。对小质数状压，共有 8 个，分别为 2,3,5,7,11,13,17,19 。设 $f_{S1,S2}$ 表示第一个人选择的数的小质数集合为 $S1$ ，第二个人选择的小质数集合为 $S2$ 。对大质数相同的数进行分组背包，要么都不选要么只能被同一个人选。时间复杂度 $O(n4^8)$ 。注意 $S1,S2$ 无交的时候状态才有效，只枚举这些状态即可。时间复杂度 $O(n3^8)$ 。

code : from_yixiuge777

设计 dp ：

【PR #1】删数

正整数序列。每次操作选择一个位置 $i\in[2,n-1]$ 使得 $a_i=\frac{a_{i-1}+a_{i+1}}{2}$ ，将 a_i 删去，之后的数顺次向前补空位。若干次操作后序列最短多少。$n\leq 300000,a_i\leq 10^9$。

考虑差分数组 d_i。操作即为相邻的数合并为两倍。

按照符号和 $\frac{d_i}{lowbit(d_i)}$ 分段分别做。不妨设所有数都是 2 的幂次，f_i 表示前 i 个数的答案，递推预处理出 $g_{i,j}$ 表示以 i 为右端点合并出 2^j 时的左端点。枚举 i 和前面合并成了什么转移。时间复杂度 $O(n\log v)$。

code : from_yixiuge777

随机数生成器

有 n 个数，每个数是一个 $[1,x]$ 之内的随机数。q 次询问，每次询问给出 $[l_i,r_i]$，计算区间内的最小值。最终测试结果是所有询问的最大值。求测试结果期望多少。对 998244353 取模。$n,k,q\le 2000$ 。

枚举答案，求出对应的概率，求概率可以求方案数，这样就把期望转换成了方案数的计数。求某一答案对应的方案数比较困难，我们可以求答案小于等于某个数的方案数，然后差分。小于等于答案的数字为0，大于的数为1，区间的限制等价于区间内必须有一个0。

需要求出对于每个 j ，填 j 个 0 满足区间要求的方案数。之后求答案只需要乘上不同的系数。如果一个区间包含了另外一个区间，那大区间没有意义。处理出所有有用的区间，按照左端点排序后右端点单调。设 fl_i 表示包含 i 位置的最左边的区间， fr_i 表示包含 i 位置的最右边的区间。$f_{i,j}$ 表示当第 i 个点为0，已经选了 j 个0，并且满足了左端点在 i 位置以前的所有区间的方案数。转移方程：

$$f_{i,j}=\sum_{k<i,fr_k+1\ge fl_i} f_{k,j-1}$$

做起来是 $O(n^3)$ 的。观察到 fl,fr 都是因为区间性质单调不降的，所以转移的 k 一定来自某个区间，前缀和优化。时间复杂度 $O(n^2)$ 。

code : from_yixiuge777

[HNOI2011] 卡农

在 $1,2,...,n$ 中取 m 个互不相同的非空子集，异或起来为 0 的方案数。 $n,m\le 1000000$

设 f_i 表示取 i 个有序的子集的答案。注意题目要求的是无序子集，这样设便于后续递推。如果不考虑非空和互不相同的限制，随便选前 $i-1$ 个之后第 i 个便因为异或为 0 唯一确定了。随机选择前 $i-1$ 个的方案数是 $\binom{2^n-1}{i-1}\times(i-1)!$ 。减去第 i 个是空集的方案数，则前 $i-1$ 个异或为 0，符合答案要求，方案数为 f_{i-1} 。减去第 i 个集合与前面相同的方案数，这样把这两个相同的集合都去掉之后剩下的 $i-2$ 个集合又满足要求了。枚举相同的集合的位置和具体元素，方案数是 $f_{i-2}*(i-1)*(2^n-1-(i-2))$ 。递推式为 $f_i=\binom{2^n-1}{i-1}\times(i-1)!-f_{i-1}-f_{i-2}*(i-1)*(2^n-1-(i-2))$。无妨设状态为无序子集，转移的时候除以 i 即可。

code : from_yixiuge777

[NOI2020] 制作菜品

有 n 种原料，第 i 种原料质量为 d_i。做 m 道菜，每道菜用到的原料总质量恰好为 k ， $m\times k=\sum d_i$ 。每道菜最多使用两种原料，且使用原料的质量为正整数。给出一种做菜方案或无解。 $n\le 500,n-2\le m\le 2500,k\le 5000$ 。

设原料质量不降，即 $d_1 \leq d_2 \leq ... \leq d_n$。对于 $m = n-1$ 的情况，可以证明 $d_1 < k \leq d_1 + d_n$ ，反证法：如果 $d_1 \geq k$ ，则 $\sum d_i \geq nk > mk$ ，不符条件。如果 $d_1 + d_n < k$ ，则 $d_i < k, \sum d_i = (d_1 + d_n) + d_2 + d_3 + ... + d_{n-1} < (n-1)k$ ，不符条件。则总是可以使用 d_1, d_n 做成一分菜，而且 d_1 被使用完。这样 n 和 m 都减少 1 ，直到 $m = 1, n = 2$ 时构造成立。对于 $m \geq n$ 的情况，一定有 $d_n \geq k$ ，此时使用 d_n 独自做一道菜，便使其一步步转换为 $m = n-1$ 的情况，再处理即可。

下面对于 $m = n-2$ 的情况讨论。原料比较多，如果一道菜同时使用了两个原料，在两个原料中间连一条边，那么这个图一定不联通。换句话说，可以把这个图分成两部分，每一部分都满足 $m = n-1$ 。现在只需要找到一个划分的方法，满足 $(|N|-1) \times k = \sum d_i$。背包实现。可以使用 bitset 优化。时间复杂度 $O(\frac{n^2 k}{w})$ 。

code : from_yixiuge777

树(tree)sd一轮省集day9t1

无向图，对每个 k 求保留 k 条边图联通的方案数。 $n \leq 15, m \leq 200$。

容斥。将点划分为若干点集，钦定两两之间无边。设容斥系数为 $coef$ ， S 内部的边数为 t ,要选择的边数为 i ，则贡献答案系数为 $coef \times \binom{t}{i}$。对于一个实际拥有 x 个连通块的方案，其贡献的总系数为 $\sum_{i=1}^{x} \begin{Bmatrix} x \\ i \end{Bmatrix} coef_i$ 。我们希望他等于 $[x == 1]$。

斯特林反演：

$$\sum_{i=0}^{x} \begin{Bmatrix} x \\ i \end{Bmatrix} \begin{bmatrix} i \\ y \end{bmatrix} (-1)^{i-y} = [x = y]$$

代入 1 得到

$$\sum_{i=0}^{x} \begin{Bmatrix} x \\ i \end{Bmatrix} (i-1)!(-1)^{i-1} = [x = 1]$$

因此求得 $coef_k = (k-1)!(-1)^{k-1}$。

设 $f_{S,i,j}$ 为钦定点集为 S，划分为 i 个点集，总边数为 j 的带容斥系数的值。转移时枚举子集。时间复杂度 $O(3^n nm)$。

只要在枚举的时候认为点集有序，就自然有 $k!$ 的系数，钦定 $1 \in S_1$ 系数就会变成 $(k-1)!$ 。减少一维状态。这是有序转移和无序转移的区别。时间复杂度 $O(3^n m)$。

code : from_yixiuge777