# 基础动态规划——常见模型与技巧

4182_543_731

2024/07

# Table of Contents

# 局部限制与状态

一些众所周知的 DP 问题：

每个物品有重量 $w_i$，最多选一次，求总和不超过 $C$ 的方案数。

# 局部限制与状态

一些众所周知的 DP 问题：

每个物品有重量 $w_i$，最多选一次，求总和不超过 $C$ 的方案数。
记 $dp_{i,j}$ 表示前 $i$ 个物品选了重量和为 $j$ 的方案数，则

$$dp_{i,j} = dp_{i-1,j} + dp_{i-1,j-w_i}$$

$n$ 个元素排成一列，不能同时选相邻的两个元素，求最大总和

# 局部限制与状态

一些众所周知的 DP 问题：

每个物品有重量 $w_i$，最多选一次，求总和不超过 $C$ 的方案数。
记 $dp_{i,j}$ 表示前 $i$ 个物品选了重量和为 $j$ 的方案数，则

$$dp_{i,j} = dp_{i-1,j} + dp_{i-1,j-w_i}$$

$n$ 个元素排成一列，不能同时选相邻的两个元素，求最大总和
记 $dp_{i,0/1}$ 表示前 $i$ 个物品，第 $i$ 个是否可以选的最大总和，则

$$\begin{cases} dp_{i,1} = \max(dp_{i-1,1}, dp_{i-1,0} + v_i) \\ dp_{i,0} = dp_{i-1,1} \end{cases}$$

# 局部限制与状态

什么是 DP？这极其难以回答，但我们可以尝试总结一些共性。

- 我们可以在问题中用简单的信息描述一个局部的子问题（**状态**）。例如，在背包问题中，由于问题只考虑总和，在处理了前若干个物品后，我们只需要关心总重量，而不需要考虑具体选了哪些物品。
- 问题的限制具有一定的局部性，使得我们可以用简单的**转移**将子问题连接起来，进而得到原问题的答案。例如，在链上独立集问题中，限制只和相邻两个元素有关，因此转移时只需要再考虑下一个元素是否被选。

很多 DP 问题的关键便在于，通过分析原问题，找到合适的**状态**，然后通过合适的**转移**得到所有子问题的结果。

# 局部限制与状态

状态设计需要注意的点：在子问题中，当前记录的状态是不是足够处理问题的限制？

转移设计需要注意的点：这样的转移是否不重不漏？（计数）/是否找到了最优解？（最优化）

我们将在接下来的若干模型和例子中体会这一点。

# 序列 DP

最最常见的 DP 模型。

最常见的状态设计是前缀：记 $dp_{i,\dots}$ 表示考虑了序列前 $i$ 个元素，然后 ...序列上的很多限制是相邻的，因此我们通常可以只记录很少的状态以解决问题。

## Tree Planting(Easy)

有 $n$ 个位置，每个位置有 $a_i$ 种方式选 $0$，$b_i$ 种方式选 $1$。要求：

- 相邻两个位置不能同时选 $1$。
- 距离为 $k$ 的两个位置不能同时选 $1$。

求合法方案数。$n \leq 300, k \leq 16$

# 序列 DP

最最常见的 DP 模型。

最常见的状态设计是前缀：记 $dp_{i,\dots}$ 表示考虑了序列前 $i$ 个元素，然后 ...序列上的很多限制是相邻的，因此我们通常可以只记录很少的状态以解决问题。

另一种常见的可能性是区间：记 $dp_{l,r}$ 表示考虑区间 $[l,r]$ 内的状态，...有些时候，我们不能直接按照序列顺序考虑问题。但序列进行分裂后还是区间，因此按照其它角度通常会得到区间 DP。

相信大家都会，这里就不赘述了。我们将在综合练习中进一步考虑。

# Table of Contents

# 状态压缩

在一些情况下，DP 的状态会比较复杂，但我们实现程序的时候一般只能用数来描述状态。这种时候，我们通常可以通过一些方式把状态映射到一个数。

这更多的是一种思想：我们可以把很复杂的东西作为一个状态，而不一定总是 $dp_{i,j,k}$。

最经典的，如果状态是 $\{1, 2, \cdots, n\}$ 的一个子集，它可以被描述为一个 $n$ 位 $01$ 串，其中第 $i$ 位表示 $i$ 是否在集合中。这又可以被看成一个二进制数。通过简单的位运算，我们可以快速支持一些简单操作：求集合的交，判掉一个元素是否在集合内，求集合大小（预处理）...

在 $n$ 不大的情况下，我们完全可以把集合作为一个状态。

# 子集 DP

子集 DP 的第一种常见转移：加入下一个元素。

**[CCO2015] Artskjid**

给一张有向图，求 $1$ 到 $n$ 的最长简单路。$n \le 18$

# 子集 DP

子集 DP 的第二种常见转移：加入下一个子集。

**经典例题 2**

给定 $\{1, \cdots, n\}$ 中的一些子集是好的，求有多少种将 $\{1, 2, \cdots, n\}$ 划分为若干个好的子集的方案。$n \le 15$

当然，对于一个一般的 01 串，我们也可以用相同的位运算技巧来处理：

**Tree Planting(Easy)**

有 $n$ 个位置，每个位置有 $a_i$ 种方式选 $0$，$b_i$ 种方式选 $1$。要求：

- 相邻两个位置不能同时选 $1$。
- 距离为 $k$ 的两个位置不能同时选 $1$。

求合法方案数。$n \leq 300, k \leq 16$

一个常见的情况是二维问题：有一个宽度较小的矩形，在限制只和矩形相邻位置有关的情况下，我们可以一行一行考虑。

**[USACO06NOV] Corn Fields**

有一个 $n \times m$ 的网格，有些格子上面可以放棋子。你不能在四相邻的格子中同时放格子，求合法方案数。$n, m \leq 12$

改成八相邻：

### [NOI2001] 炮兵阵地

有一个 $n \times m$ 的网格，有些格子上面可以放棋子。两个放的棋子不能距离不超过 $2$ 且同行或同列。求最多能放多少棋子。$n \leq 100, m \leq 10$

### [USACO06NOV] Corn Fields 加强版

有一个 $n \times m$ 的网格，有些格子上面可以放棋子。你不能在四相邻的格子中同时放格子，求合法方案数。$n, m \leq 18$

# 插头 DP / 轮廓线 DP

**Rikka with K-Match(Easy)**

有一个 $n \times m$ 的网格图，边有边权，求最大权匹配的权值。

$n \le 5 \times 10^4, m \le 6$

**Tree Planting(Medium)**

有 $n$ 个位置，每个位置有 $a_i$ 种方式选 $0$，$b_i$ 种方式选 $1$。要求：

- 相邻两个位置不能同时选 $1$。
- 距离为 $k$ 的两个位置不能同时选 $1$。

求合法方案数。$n \le 150$

# 插头 DP / 轮廓线 DP

**[HDU 多校] Tree Planting**

$n \leq 300$

对于更一般的复杂状态，我们也可以把它强行表示下来。这时通常不能表示得很简单，但多数时候这里不会卡时间，可以手动维护映射/强行 map。

**[NOI2007] 生成树计数**

$n$ 个点，两个点 $i, j$ 间有边当且仅当 $|i-j| \leq k$，求生成树数量。$n \leq 10^{15}, k \leq 5$。

### [HDU 多校] Yinyang

给一个 $n \times m$ 的网格，把每个格子染成黑白之一，满足如下条件：

- 黑色格子四连通
- 白色格子四连通
- 任意一个 $2 \times 2$ 的连续子矩阵不同色

现在给定部分颜色，求合法地染剩下部分的方案数。$nm \le 100$

# Table of Contents

# 数位 DP

$$
\begin{array}{cccccc}
 & & & 1 & 0 & 1 \\
+ & {}_{1} & 1_{1} & 1 & 0_{1} & 1 \\
\hline
 & 1 & 0 & 0 & 1 & 0
\end{array}
$$

如图所示，有很多操作都可以是按位做的：加减法，比较，位运算，...~~当然还有问题要求你考虑数位~~

这些操作在按位时都有着很好的局部性：加减法只需要从后往前，记录后面进位了多少；比较最常见的方式是从大到小一位一位比；位运算自然只考虑这一位。

在这种时候，我们可以一位一位地填数，然后记录考虑了这些位后上面这些问题需要的状态。常见的状态设计即为：填了高/低 $i$ 位，当前进位多少/比较关系如何/...

从高到低还是从低到高？根据问题考虑。一般的关键点在于加减法和比较在两种顺序下的表现。对于比较：

- 在从低到高的情况下，比较需要记录两个后缀的大小关系。如果限制是 $a \le b$，那么只需要记录 $a$ 的后缀是否 $\le b$ 的后缀。
- 在从高到低的情况下，前缀比出了就可以直接确定关系。假设限制是 $a \le b$，则如果前缀比出了 $a > b$ 就可以直接跳过，只需要记录前缀是已经比出来 $a < b$，还是不确定。

看似效率没有差别，但从高到低有一个好处：如果已经比出来了，就不会变回去，因此下面的转移只有三种，而上面的有四种。在有多个数的情况下，结合状态压缩技巧可能可以做到类似 $3^m$ 和 $4^m$ 的区别。通常我们会从高往低做。

### [SCOI2009] windy 数

求 $[l, r]$ 中有多少整数 $x$ 满足：$x$ 的十进制表示下相邻两数位差不小于 $2$。

$r \leq 2 \times 10^9$

# 数位 DP

「PA 2020」Liczba Potyczkowa

求 $[l, r]$ 中有多少整数 $x$ 满足其十进制表示中不存在 $0$，且 $x$ 被其十进制表示中每种出现过的数整除。

$l, r \leq 10^{18}$

# 数位 DP

从高到低还是从低到高？根据问题考虑。一般的关键点在于加减法和比较在两种顺序下的表现。对于加法：

- 在从低到高的情况下，加减法是非常简单的：状态只需要记录向前进位了多少，转移也很直接。
- 从高到低的加减法则稍微复杂一点：状态可以记作“如果后面进位了 $k$，则前面这样的方案数”。转移时枚举下一位情况，算出下一位又应该进位多少。

效率上其实不存在差别，前者稍微直观。

## [CF 1710C] XOR Triangle

求有多少组非负整数 $a, b, c$ 满足：

- $a, b, c \le n$
- 记 $x = a \oplus b, y = b \oplus c, z = c \oplus a$，则 $x + y > z, y + z > x, z + x > y$

$n \le 2^{2\times 10^5}$，输入为二进制

# 数位 DP

**Lotus(Easy)**

求有多少组非负整数 $a_i$ 满足 $\sum a_i 2^i \leq 2^n$。

$n \leq 500$

## 「THUPC 2021」游戏

考虑到您可能不会博弈，问题相当于：求有多少组非负整数 $a_1, \cdots, a_m$ 满足

- $\sum a_i = n$
- $\oplus a_i \neq 0$
- $a_i \leq l_i$

$m \leq 10, n \leq 10^{18}$

# 数位 DP

当然更一般的情况下，从高到低确有其优势：

## 分形图

求有多少组非负整数 $a_1, \cdots, a_m$ 满足

- $a_i \in [l_i, r_i]$
- 对于每一位，这 $m$ 个数在这一位上的取值必须取某些组合（组合数量不超过 $2^m$）

$m \leq 11, V \leq 2^{60}$

# Table of Contents

# 树形 DP

树也是一个极其常见的 DP 模型。它有着最自然的子问题形式：以一点 $u$ 为根的子树。以 $u$ 为根的子树与剩余部分相邻的点只有根 $u$，因此我们通常只需要记录一些与子树根 $u$ 有关的状态，就可以维护足够的信息。

**没有上司的舞会**

树上每个点有点权，求最大权独立集。$n \le 10^6$

考虑填了一个子树后我们需要记住什么状态。因为子树内只有根 $u$ 和外面相连，因此只需要记录 $u$ 的状态。

记 $dp_{u,0/1}$ 表示填了 $u$ 的子树，$u$ 是否被选择时的最优答案，则显然

$$\begin{cases} dp_{u,1} = w_u + \sum_{v\in son_u} dp_{v,0} \\ dp_{u,0} = \sum_{v\in son_u} \max(dp_{v,0}, dp_{v,1}) \end{cases}$$

树形 DP 的状态设计要点：考虑了整个子树之后，在上面的问题中要记录多少信息?
树形 DP 的转移：先合并所有子树的信息，然后考虑根向上的边。

**小练习**

给一棵 $n$ 个点的树，删掉一些边，然后在剩余每个连通块内选一个点，求方案数。
$n \leq 10^6$

# 树形背包

**[51nod 1353] 树**

给一棵 $n$ 个点的树，删掉一些边，使得剩余每个连通块大小不超过 $k$。求方案数。$n \leq 5000$

# 树形背包

**树 v2**

给一棵 $n$ 个点的树，删掉一些边，使得剩余每个连通块大小不超过 $k$。求方案数。

$n \leq 10^5, k \leq 300$

# 换根 DP

**[CF 1324F] Maximum White Subtree**

给一棵树，点权为 $\pm 1$。对每个点 $u$，求包含 $u$ 的连通块点权和最大值。$n \leq 10^6$

不一定所有树形 DP 都是从下往上的。比如上面换根 DP 的例子，比如

**某个题的某一步**

给一棵有根树，点有点权 $v_i$，对于每个叶子 $u$，记 $path_u$ 表示 $u$ 到根的路径，求出 $\prod_{x \notin path_u} v_x$，答案模**合数** $10^9 + 2022$。

在更多的时候，状态设计不是直接的：树的性质非常好，因此我们通常可以推出很多性质，然后再根据这些性质来设计状态。推性质可以非常、非常、非常难。

**[CF 1032F] Vasya and Maximum Matching**

给一棵树，你可以任意删一些边。求有多少种删边方式使得最大匹配唯一。$n \leq 3 \times 10^5$

# 组合意义 (Intro)

## 一道题

给一棵 $n$ 个点的树，你有 $2^{n-1}$ 种方式删掉一些边，对所有方式求和最后得到的每个连通块大小乘积。

$n \leq 10^6$

# Table of Contents

# 综合练习

你已经学会了基础的 DP 模型，让我们来尝试一点基础的应用。

hint 1：在很多时候，我们需要找到正确的角度去分析问题、设计状态。

# 「PA 2019」Muzyka pop

给一个长度为 $n$ 的序列 $v$ 和 $m$，你需要找到一个长度为 $n$ 的整数序列 $a$ 使得 $0 \le a_1 \le a_2 \le \cdots \le a_n \le m$，最大化 $\sum_{i=1}^{n} popcount(a_i) * v_i$，权值可能为负数。

$n \le 200, m \le 10^{18}$

# [hdu 6710] Kaguya

有一个二分图，两侧分别有 $n, m$ 个点，中间每条边有 $1/2$ 概率存在。
求左边一个点到右边一个点的期望距离。(如果不连通，定义距离为 $0$)
多组数据，$T, n, m \leq 30$

# [NOIP2017] 宝藏

给一张 $n$ 个点的图，选择一棵生成树，再选择一个根 $u$。记一条边的深度为它到根经过的其它边数量加一，最小化

$$\sum_{e \in \text{Spanning Tree}} w_e * dep_e$$

$n \leq 12$

在任意时刻，你可以花费 $k$ 的代价进行一次强度为 $k$ 的攻击。

有 $n$ 个外星人，第 $i$ 个外星人要求你在时间段 $[l_i, r_i]$ 内至少进行一次强度不小于 $c_i$ 的攻击。

求最小总代价。

$n \le 300$

# [CERC2014] Parades

给一棵 $n$ 个点的树，保证每个点度数不超过 $d$。

给 $m$ 条路径，选出尽量多的路径，满足选出的路径不使用相同的边（可以使用相同的点）。求最多能选多少路径。

$n \leq 1000, d \leq 10$

# 「PA 2019」Desant

给一个 $n$ 阶排列，对于每个 $k=1,2,\cdots,n$，解决如下问题：

你需要选一个长度为 $k$ 的子序列，最小化逆序对数量。求最小的逆序对数量和达成这个值的方案数。

$n \le 40, 6s$

# Thanks!

如果您是这些 DP 模型的初学者，那您更应该多加练习相关题目（这里讲到的题目只是很少的例子）以更加熟练。