2026 年寒假讲题

浅谈 DP 及其优化

DrAlfred

上海市曹杨第二中学 maze.size()

2026 年 02 月 08 日

序言

- 大家能来到这里说明肯定对 DP 有一定的了解，本节课主要是教大家一些状态设计和优化手法。
- 讲者很菜。Luogu: 583610，QQ: 229882561，欢迎联系交流。
- 大家需要课件和配套资料可以访问 https://github.com/CYEZOI/2026-Winter-OI 或者 https://share.alfredbao.cn/。
- 接下来的 3h 祝大家 GL & HF，有问题随时提问！

如何定义 DP

- 我讲之前，大家可以先说说自己的想法。

如何定义 DP

- DP 实际上是一个分步生成解的自动机结构。我们通过分步生成解把指数级搜索空间压缩到多项式级别，找到解的某种权值经过某种运算后的结果。
- 这种情况下的 DP 大致有三类：最优化、计数、判定。在所有合法解不漏的前提下，它们需要满足的关键条件是：最优子结构、不计重、无后效性。
- 广义上来说，习惯于将大部分用递推解决问题的方法都称作 DP。
- 如果可以找到一种生成方式，能恰好（逐步）刻画出所有满足条件的解，并且中途为了判定符合条件以及为了辅助求出权值所记录的信息是局部的或可能性较少的，那就有机会使用 DP。

括号序列

生成思路

括号序列的生成思路一般有以下几种：

1. 直接按下标，记录当前多的 (数。
2. 按外层配对括号 (...)(...)...(...) 拆开。
3. 拆第一对配对括号，剩下视作整体 (...)...。
4. 视作折线，进行容斥。

- 方式 2、3 是直接对着定义 DP，3 可以理解成对括号树三度化再 DP
- 方式 2 在最优化问题下可以随便选一个断点拆

括号序列：类括号序列

01 序列技巧

如果有一些 0 和一些 1，每个都有权值，要选一个子集 01 数量相同，最优化权值相关某个东西：

- 可以把元素随机打乱再 DP
- 记录当前 0 比 1 多多少
- 这一维可以只开到 $\Theta(\sqrt{n})$

排列：基本生成顺序

这一节侧重于排列未知的情况。参照 YeahPotato 的课件，最基本的排列生成顺序如下：

	预定（绝对）	插入（相对）
按下标	从左往右逐一确定值	从左往右逐一确定当前值在前缀中排第几大
按值	从小到大逐一确定位置	从小到大逐一插入排列

排列的关键是找到一个生成顺序，使得已生成部分的数不需要逐个记录，而只需抽象成少量的几个量。

排列：典型应用

应用类型

1. 只对一个单调子序列 DP，剩余部分直接用组合数确定
   - 属于按值、断点
2. 基于笛卡尔树或类似笛卡尔树的分治结构
   - 合并时确定相对大小，乘组合数
   - 属于按值、预定、单步

确定了主要元素（某个单调子序列），剩余元素必须出现在某个主要元素前或后时：

- 为了使状态维数小，就对主要元素 DP
- 剩余元素的位置在对应的主要元素被 DP 到时用数学方法确定
- 限制方向与 DP 方向相同则用预定法，否则用插入法

背包：基础技巧

基础背包技巧

1. 多重背包的二进制拆分和同余转移
2. 换维：价值较小时可以定义状态为得到 i 的价值至少要多少重量
3. 物品大小和有限制时只有根号种物品
4. 无序整数拆分相关：
   - "全体 $+1$ & 新开一个" 的转移方式
   - 按下标 DP 总状态数为 $n^2 \ln n$
5. 从 GF（生成函数）角度考虑
6. 同余最短路

背包：高级技巧（仅用于最优化）

完全背包的倍增方法

当容量 T 很大，最大重量 W 较小时：

- 通过将解按重量排序交替放，可证明一定能将解拆成两个总重量差 $\leq W$ 的部分
- 求出 $[T-W, T+W]$ 的最优解可以 $O(W^2)$ 递推到 $[2T-W, 2T+W]$
- 复杂度：$O(nT) \to O(W^2 \log T)$

背包：高级技巧（仅用于最优化）

背包的贪心性质

按性价比贪心选直到第一次遇到无法选的物品就停止，设选出的物品集合为 G，最优解为 O：

- 性质：存在 O 使 $|G \oplus O| \leq 2W$
- 只需一个状态数为 $4W^2$ 的背包即可
- 复杂度：$O(nT) \to O(nW^2)$

状压 DP：优化技巧

状压 DP 的难点主要在两块：刻画生成过程和优化。

meet-in-the-middle

如果转移时要考虑两个数之间的某个位运算，可以两者分别枚举一半。

几个经典的转移形式（$\sqcup$ 表示不交并）

1. $g_i = \sum_{j \sqcup k = i} f_j g_k$：半在线子集卷积，$O(n^2 2^n)$
2. $g_i = T_i\left(\sum_{j \sqcup k = i} f_j g_k\right)$：按 $|i|$ 从小到大处理，$O(n^2 2^n)$
3. $g_i = \sum_{j \sqcup k = i, h(j) = h(i)} f_j g_k$：全集的无序划分，$O(n^2 2^n)$
4. $g_i = T_i\left(\sum_{j \subset i} g_j\right)$：按最高位 01 分治，$O(n 2^n)$

数位 DP：基本思路与技巧

全体 $+x$ 的最优化问题

考虑逐位 DP，记录进位的分界线：

- 如果当前确定的是低 i 位
- 一定是所有数按低 i 位排序后的一个后缀进位
- 转移时先基数排序，再枚举进位分界，得到下一位每种数占的区间

重要模型

考虑函数 $f(S) = \{\lfloor x/2 \rfloor, \lceil x/2 \rceil \mid x \in S\}$：

- $f^{(i)}(\{x\})$ 的大小至多为 2
- $f^{(i)}(\{x\}) = \{\lfloor x/2^i \rfloor, \lceil x/2^i \rceil\}$

树形 DP：设计思路与技巧

基本设计思路

- 考虑加一条边和合并两个儿子
- 把状态和转移定出来

比较初级的技巧

1. 换根 DP
2. 多儿子选择时的反悔技巧
3. 树上背包
   - 如果强制大小与 k 取 min，时间是 $O(nk)$ 的

另一种 DP 方式

在 DFS 序上跳

树形 DP：树上背包复杂度证明

$O(nk)$ 复杂度证明

采用树分块的思路：

- 子树 $\mathrm{siz} \le k$ 的点称为小点，剩余称为大点
- 小点：每个极大的小点子树都是 $O(\mathrm{siz}^2)$，由均值不等式至多 $O(nk)$
- 小点顶部转移给大点：$O(\mathrm{siz} \cdot k)$，共 $O(nk)$
- 大点之间：两个大点合并为 $O(k^2)$，合并共发生 $O(n/k)$ 次，也是 $O(nk)$

开始优化之前

先对自己进行一些灵魂拷问：

- 你找全这个结构的性质了吗？
- 你找对该对着 DP 的对象了吗？有没有可以解构的地方？
- 你设计的 DP 状态已经尽可能少了吗？

主要思考以下几个方面：

- 寻找可以直接从题目条件导出，或与题目条件等价的条件；
- 模拟（包括打表）小样例、特殊样例找到简化的方向和规律；
- 另外，如果解是一个过程，有时倒推一些必要的东西会比较容易。
- 联想一些与题目条件相关或类似的模型，尝试套用。

毕竟，最好的优化是不用其他优化。

开始优化之前

QOJ 16228 Sum of Three Inversions

给定长度 n，三个常数 x, y, k，计数下列序列三元组 (A, B, C) 的个数：

- A, B, C 均为长度为 n 的序列；
- $\forall i \in [1, n], (A_i, B_i, C_i)$ 是 $(1, 2, 3)$ 的一个排列。
- A 中恰好有 x 个 1，y 个 2。
- 序列 A, B, C 中的逆序对总数为 k。

$n \leq 50$，答案对给定的常数 m 取模。

尝试写出所有你认识到的性质。你能做到的最优复杂度是什么？

开始优化之前

- 可以直接对 $D_i = (A_i, B_i, C_i)$ 计数。发现大多数长度为 3 的排列前后之间可以贡献的逆序对数量为 1，相同的排列之间贡献为 0，有一些排列前后之间可以贡献 2 个逆序对；
- 可设 $f_{c_1,c_2,c_3,i}$ 表示现在分别有 c_1, c_2, c_3 个组内的第一个为 1 排列，第一个为 2 排列，第一个为 3 排列，当前逆序对贡献为 i 的方案数。只有组内可能贡献 2 或 0 个逆序对。考虑设定基准贡献为一个逆序对，那么特殊的对之间的贡献就变成 $+1$ 或 -1 了，而组间没有贡献，两组内的结构是相同的。
- 转移是容易的，然后考虑把第一组插入到第二组里面。最终答案为：

$$\sum_{a=0}^{x}\sum_{b=0}^{y}\sum_{c=0}^{z}\sum_{i=-\frac{n(n-1)}{2}}^{\frac{n(n-1)}{2}}\binom{n}{a+b+c}f_{a,b,c,i}f_{x-a,y-b,z-c,k-i}$$

开始优化之前

- 时空复杂度为 $O(n^5)$，真能跑满吗？
- 分析一下，$c_1 + c_2 + c_3 \le n$，取值方案只有 $\binom{n+3}{3}$ 种。再加上 $-\binom{c_1+c_2+c_3}{2} \le i \le \binom{c_1+c_2+c_3}{2}$。实测状态数大概只有 10^7 左右。
- 有时，真正可能 DP 到的状态远少于 DP 状态每一维最大值之积。可能使用记搜简化代码。
- Fun fact: 这个题 maze.size() 的记忆化搜索实现跑了 1695ms，而我校另一支队伍 Be over the moon 李神的精细实现跑了 5ms。

决策单调性：定义

决策单调性

在动态规划时，若转移时对于一个状态是单点对单点转移，且令 k_i 表示状态 i 的最优决策点，有 $\forall i < j$ 有 $k_i \leq k_j$，则称这个动态规划的过程具有决策单调性。

判定方式

1. 打表：使用朴素 DP 求出每个点的最优决策点，对于每一个点的最优决策点进行比较
2. 四边形不等式法则与区间包含单调性

决策单调性：四边形不等式

四边形不等式

对于函数 $w(i,j)$，$\forall p_1 \le p_2 \le p_3 \le p_4$，有如下式子：

$$w(p_1,p_3)+w(p_2,p_4) \le w(p_1,p_4)+w(p_2,p_3)$$

则我们称函数 $w(i,j)$ 满足四边形不等式。
记忆为：交叉小于等于包含。

区间包含单调性

如果对于函数 $w(i,j)$，如果 $\forall a \le b \le c \le d$，有 $w(b,c) \le w(a,d)$，则称函数 $w(i,j)$ 满足区间包含单调性。

决策单调性：1D/1D 动态规划

标准形式

$$f[i] = \min\{f[j] + w(j,i)\}(0 \le j < i)$$

定理

如果这个式子中 $w(j,i)$ 满足四边形不等式，则整个动态规划的过程具有决策单调性。

优化效果

使用决策单调性可以将复杂度从 $O(n^2)$ 优化至 $O(n \log n)$。

决策单调性：优化方法

二分单调队列

- 每一个状态都有一个决策表，存储当前的最优决策
- 当状态 i 转移完成后，找到一个最小的位置，使这个位置由 i 转移比目前决策表的转移更优
- 根据决策单调性，该位置之后的位置，其决策表都更新为 i
- 这些段形容成三元组 $s(x, l, r)$，表示 $[l, r]$ 的状态最优转移点为 x

分治优化

如果状态间转移无依赖，可以使用分治优化，时间复杂度仍为 $O(n \log n)$。

斜率优化

适用条件

转移形式类似：

$$f[i] = \min_{j<i}\{f[j] + a[i] \cdot b[j] + c[i]\}$$

可以理解为在平面上维护一个凸包，每次查询最优决策点。

基本思路

1. 将转移方程改写成斜率的形式
2. 维护一个下凸壳或上凸壳
3. 利用单调性优化查询

复杂度为 $O(n)$ 或 $O(n \log n)$，取决于斜率和决策点是否单调

WQS 二分

适用问题

- 有一个限制维度很难处理（如恰好选 k 个）
- 答案关于这一维具有凸性

基本思路

1. 去掉" 恰好选 k 个" 的限制
2. 二分一个代价 c，每选一个就额外付出代价 c
3. 找到使得恰好选 k 个的代价 c
4. 原问题的答案就是 $f[n] - k \cdot c$

关键

需要证明答案关于选择数量的凸性！

状态优化：换元与合并

下标换元

令 g_i 表示 $f_{t(i)}$ 然后对 g 做 DP：

- 能方便一些转移优化
- 常能暴露出一些不必记或不必分开记的状态
- 触发状态数的自简化或状态合并

状态合并

在初步设计 DP 时，可能会设计一些含义较好理解的，但实际上有些信息没有必要区分的 DP 状态，可以在进一步分析时合并。

最优化专用优化

最优性去状态

利用一些性质去除不必要记的状态：

- 这个状态不可能生成最优解
- 或有多条路线可以生成最优解，去掉其中的部分

最优性换维

交换一维状态和所记值：

- 将求" 某种情况下最优的……值"
- 转为求" 要达到最优值为……，某维的值至少/多要达到多少"
- 注意：要刻意证明最优子结构！

数据结构优化

常见的数据结构优化

1. 线段树：维护区间最值、区间和等
2. 单调队列/单调栈：维护滑动窗口最值
3. 平衡树：动态维护有序序列
4. 树状数组：维护前缀和、区间和

注意

数据结构优化往往是在其他优化之后的最后一步，不要一开始就想着用数据结构硬上。

DP 优化：一般流程

优化流程

1. 分析模型：找全性质，确定 DP 对象
2. 设计状态：尽可能少的状态维度
3. 写出方程：清晰的转移方程
4. 观察方程：
   - 组合意义
   - 转移方程的特征（四边形不等式、凸性等）
   - 转移图的特点
5. 选择优化：
   - 状态优化：换元、合并、去除
   - 转移优化：决策单调性、斜率优化、WQS 二分
   - 数据结构：线段树、单调队列等

致谢

- 感谢曹杨二中 OI 教练组提供的本次交流机会；
- 感谢同学们三个小时的认真听讲；
- 感谢 YeahPotato 的优秀文章《dp 题方法总汇》提供的灵感。
- 感谢我的同学兼 XCPC 队友 JoeyJ 提供的优质例题。
- 希望大家都能成为 DP 高手。/bx /bx /bx