广义 B-F 算法例题与解答

whoami

October 18, 2022

广义 B-F 算法

- General Brute Force Algorithm
- 由 Brute 与 Force 在 1926 年 8 月发表
- 可以优化各类问题求解
- 各个领域均有广泛应用

是假的

- 上一页均是胡说八道
- 而且这个 PPT 很长
- 讲的还不是这题的标准解法
- 没兴趣的同学可以先睡一觉

小水獭与电影院 (Easy)

- 给定两个正整数 n, m
- 计算满足每行每列均无连续 1 的 $n \times m$ 的 01-矩阵的数量
- 答案对 2^{32} 取模
- T 组数据
- $1 \le nm \le 400$， $1 \le T \le 10$
- 时间限制： 6s

名词约定

- 不妨称「满足每行每列均无连续 1 的 $n \times m$ 的 01-矩阵」为合法的 01-矩阵
- 这并没有不满足条件就违法的意思

什么是 OEIS？

- The On-Line Encyclopedia of Integer Sequences ®
- 整数序列在线百科全书
- 可以请 OEIS 帮忙找找规律

复制 & 粘贴？

- OEIS 是个好东西
- $n + m \leq 50$ 的答案在 OEIS 的数列 A089934 可以找到
- $1 \leq n \leq 7$ 对应的数列在 crossrefs 里有链接
- 相关文献在 comments 里有链接
- 数据拎下来好像还差好多项，输了（

有趣的事实们

- 记合法的 $n \times m$ 的 01-矩阵数量为 $A(n, m)$（不必取模）
- 以下是一个有趣的事实：

$$A(1, n) = fib_{n+1}$$

- 以下是一些没那么有趣的事实：

$$\begin{aligned}
A(2, n) &= 2A(2, n-1) + A(2, n-2) \\
A(3, n) &= 2A(3, n-1) + 6A(3, n-2) - A(3, n-4) \\
A(4, n) &= 4A(4, n-1) + 9A(4, n-2) \\
&\quad - 5A(4, n-3) - 4A(4, n-4) + A(4, n-5)
\end{aligned}$$

常系数线性齐次递推

- 设 $\{f_n\}$ 为一数列，$c_1, \dots c_k$ 为常数且 $c_k \neq 0$，则称以下递推关系：

$$f_n = \sum_{i=1}^{k} c_i f_{n-i}$$

为 k 阶常系数线性齐次递推关系

- 不难发现 $\{A(1,n)\}, \dots, \{A(4,n)\}$ 都满足常系数线性齐次递推关系
- 哇！那是不是求出来递推的系数就赢了？

又输了

- 如果递推阶数比较小说不准还可以试试
- 但是从 OEIS A089935 看起来
- 递推阶数增长起来有点 Fibonacci 数列的意思
- 这种方法是行不通的

Part O 总结

- OEIS
- 常系数线性齐次递推

一些浅显的观察

- 重要的话说两遍：
  - 记合法的 $n \times m$ 的 01-矩阵数量为 $A(n, m)$（不必取模）
- $A(n, m) = A(m, n)$
  - 因为合法的 $n \times m$ 矩阵转置后仍满足条件
- $\min\{n, m\} \leq 20$
  - 因为 $nm \leq 400$
- 不妨假设 $m \leq 20 \leq n$

行状态

- 重要的话说至少两遍:
  - 记合法的 $n \times m$ 的 01-矩阵数量为 $A(n, m)$（不必取模）
  - 不妨假设 $m \le 20 \le n$
- 考虑按行为单位的动态规划
- $n \times m$ 的 01-矩阵中一行的 m 个 $0/1$ 作为一个行状态
- 行状态可以与 $[0, 2^m)$ 中的整数一一对应:

$$0\ 1\ 1\ 0\ 1\ 0\ 1 \rightarrow 2^5 + 2^4 + 2^2 + 2^0$$

- 接下来我们不再区分行状态与对应的整数

合法行状态

- 重要的话说三遍：
  - 不妨假设 $m \leq 20 \leq n$
- 我们只关心不出现连续 1 的行状态
- 不妨称之为合法行状态
- 合法行状态的数量是 $A(1,m) = fib_{m+1}$
  - 这是因为有如下递推关系：

$$A(1,m) = \underset{\substack{\uparrow \\ \text{xxx...x0}}}{A(1,m-1)} + \underset{\substack{\uparrow \\ \text{xxx...01}}}{A(1,m-2)}$$

  - 并且有边界条件 $A(1,0) = 1$，$A(1,1) = 2$

朴素动态规划

- 记 $f_{i,s}$ 为满足第 i 行的行状态为 s 的 $i \times m$ 的合法 01-矩阵的数量
- 记合法行状态构成的集合为 V_s
- 有以下状态转移方程:

$$f_{i+1,s} = \sum_{0 \le t < 2^m} [t \in V_s][s \,\&\, t = 0] f_{i,t}$$

其中 $\&$ 是二进制与，$[P]$ 表示命题 P 成立取 1 否则取 0

朴素动态规划

- 初始条件为:

$$f_{0,0} = 1$$

- 所求答案为:

$$A(n, m) = \sum_{0 \le t < 2^m} [t \in V_s] f_{n,t}$$

朴素动态规划的结果

- 然后会发现喜提 TLE
- 为什么呢？让我们来算一算复杂度
- 哇！时间复杂度居然是 $O(Tn4^m)$
- 交给天河一号来跑应该能 AC
- 可惜 accoding 的评测机不是天河一号

Part I 总结

- $nm \le 400 \implies \min\{n, m\} \le 20$
- 状态压缩动态规划
  - 将多个状态编码为一个状态的动态规划

不要暴力

- 我们需要来优化优化 DP
- 回顾状态转移方程，DP 过程中枚举了很多无效状态
- 预处理所有合法行状态
- 将状态转移方程改写为：

$$f_{i+1,s} = \sum_{t \in V_s} [s \,\&\, t = 0] f_{i,t}$$

- 时间复杂度为 $O(Tn \cdot fib_{m+1}^2)$

但是也优雅不起来

- 还可以进一步减少状态数量吗？
- 将矩阵左右镜像对称不改变合法性
- 记 $rev(s)$ 为 s 在 m 位二进制下的倒序

$$rev(0\ 1\ 1\ 0\ 1\ 0\ 1) = 1\ 0\ 1\ 0\ 1\ 1\ 0$$

- 这说明 $f_{i,s} = f_{i,rev(s)}$

有人在看标题吗

- 可以将 s 与 $rev(s)$ 合并为同一个状态
- 状态数量下降到 $\frac{fib_{m+1}+fib_{m/2+1}}{2}$ 级别
- 时间复杂度为 $O(\frac{1}{w}Tn \cdot fib_{m+1}^2)$，其中 $w=4$
- 然后会发现喜提 TLE

那有人在看 PPT 吗

- 多组数据好像有点假？
- 给十组 $n = m = 20$ 可以只算一遍
- 一次 DP 可以算出来所有 m 相同的答案
- 时间复杂度为 $O(T + \frac{1}{w} n \cdot fib_{m+1}^2)$，其中 $w = 4$
- 然后会发现喜提 TLE

时间复杂度分析

- 上一页中声称时间复杂度为 $O(T + \frac{1}{w} n \cdot fib_{m+1}^2)$，其中 $w = 4$
- 有没有人发现 T 换位置了？为什么？
- 记 $u_m = \frac{400}{m} \cdot fib_{m+1}^2$ 有：

$$\frac{u_m}{u_{m-1}} \sim \frac{m-1}{m}\phi^2 \geq \frac{1}{2}\phi^2 = \mu \approx 1.3$$

- 粗略地估计有：

$$\sum_m u_m \leq u_{m_{\max}} \sum_{n=0}^{+\infty} \mu^{-n}$$

- 右侧无穷级数收敛于常数 $\frac{\mu}{\mu-1} \approx 4.236$

Part II 总结

- DP 时仅保留有效状态
- 对称性减少 DP 状态
- 预处理答案 / 记录已计算的答案
- 时间复杂度分析

一分为二

- 本节将说明以下事实:

$$A(n,m) = f_{n+1,0} = \sum_{t \in V_s} f_{j,t} f_{n+1-j,t}$$

其中 $1 \leq j \leq n$

  - 为什么?
  - 前 j 行与后 $n-j+1$ 行在第 j 行相交
  - $(n+1-j) \times m$ 的矩阵上下镜像后不改变合法性
  - $f_{n+1-j,t}$ 等于第一行状态为 t 的 $(n-j+1) \times m$ 合法矩阵数量
  - 对应行状态相乘后求和即为答案
  - 可能解释起来有点抽象

合二为一

- 这意味着只需求出 $f_{\lfloor n/2 \rfloor, s}$ 和 $f_{\lceil n/2 \rceil, s}$ 即可求出答案
- 时间复杂度优化为 $O(T + \frac{1}{w} n \cdot fib_{m+1}^2)$，其中 $w = 8$
- 怎么一直在优化常数
- 还是来看看为什么 DP 值能拼起来吧

一类动态规划

- 回顾动态规划的状态转移方程：

$$f_{i+1,s} = \sum_{t \in V_s} [s \,\&\, t = 0] f_{i,t}$$

- 不妨记 $V_s = \{s_1 = 0, s_2, \ldots, s_k\}$
- 将 $f_{i,s}$ 看作向量 $f_i \in \mathrm{R}^k$：

$$f_i = (f_{i,s_1}, f_{i,s_2}, \ldots, f_{i,s_k})^\top$$

- 并设常数矩阵 $A \in \mathrm{R}^{k\times k}$ 满足 $A_{ij} = [s_i \,\&\, s_j = 0]$
- 则状态转移方程可写成如下形式：

$$f_{i+1} = Af_i$$

DP 的转移是什么？

- 在这一类动态规划中，一轮转移相当于在原向量上左乘转移矩阵
- 等价于作用一次转移矩阵 A 对应的线性变换 $\mathcal{A}$
- 计算 n^k 可以快速幂，计算 $A^k f_0$ 也可以快速幂
- 在状态数少的时候可以加速 DP 计算
- 但需要指出只有少数 DP 的转移矩阵是常数矩阵

答案是什么？

- 边界条件是

$$f_0 = (1, 0, \dots, 0)^\top$$

- 本题所求的答案恰好是 $f_{n+1,0}$
  - 这是由于合法行状态均可转移到 0
- 改写一下式子：

$$A(n, m) = f_{n+1,0} = (1, 0, \dots, 0)A^{n+1}f_0 = A^{n+1}_{11}$$

$$f_{i,s_t} = e_t A^i f_0 = A^i_{t1}$$

线性代数 袋鼠现形

- 相信各位一定还记得线性代数中的以下事实:
  - $A^{p+q} = A^p A^q$
  - $(AB)_{ij} = \sum_k A_{ik} B_{kj}$
- 注意到本题 DP 的转移矩阵 A 对称即知:

$$A_{11}^{n+1} = \sum_t A_{1t}^j A_{t1}^{n+1-j} = \sum_t A_{1t}^j A_{1t}^{n+1-j} = \sum_t f_{j,s_t} f_{n+1-j,s_t}$$

- 从线性代数的角度下得到结论是自然的

* 番外 从特征多项式

- 依 Cayley-Hamilton 定理，矩阵 A 的特征多项式

$$\varphi(\lambda) = \lambda^n + c_1\lambda^{n-1} + \cdots + c_{n-1}\lambda + c_n$$

是 A 的一个零化多项式，也即：

$$\varphi(A) = A^n + c_1 A^{n-1} + \cdots + c_{n-1} A + c_n I = O$$

- 移项即有：

$$A^n = -c_1 A^{n-1} - \cdots - c_{n-1} A - c_n I$$

* 番外　到常系数线性齐次递推

- 等式两侧左乘 $A^{k+1}\ (k \geq 1)$ 有:

$$A^{n+k+1} = -c_1 A^{n+k} - \cdots - c_{n-1} A^{k+2} - c_n A^{k+1}$$

- 取左上角元素即有:

$$A_{11}^{n+k+1} = -c_1 A_{11}^{n+k} - \cdots - c_{n-1} A_{11}^{k+2} - c_n A_{11}^{k+1}$$

- 令 A 取转移矩阵立即得到:

$$A(n+k,m) = -c_1 A(n+k-1,m) - \cdots - c_n A(k,m)$$

- 这说明 $\{A(n,*)\}$ 满足常系数线性递推关系

回归正题

- 此时已经可以 AC 本题
- 最终时间复杂度为 $O(T+\frac{1}{w}n\cdot fib_{m+1}^2)$，其中 $w=8$
- 堆砌的优化并没有改变时间复杂度的阶
- 仅仅减少了计算量的常系数
  - 这也是本文标题为广义 B-F 算法例题与解答的原因（
- 还可采取滚动数组等方式常数优化
- 但本文并不是本题的标准解法（

Part III 总结

- Meet in the middle
- * 矩阵快速幂优化 DP
- * 常系数线性齐次递推优化 DP
- 常数优化

Thanks!

- Q & A