组合计数中的递推问题

Elegia
李白天

清华大学, 交叉信息研究院

2024 年 2 月 2 日

今天讲什么?

- 组合计数明明处处都有递推, 所以几乎什么都可以讲.

今天讲什么?

- 组合计数明明处处都有递推, 所以几乎什么都可以讲.
- 基于生成函数的处理手法, 部分经典算法, 以及它们一些 OI 之外的故事.

今天讲什么?

- 组合计数明明处处都有递推, 所以几乎什么都可以讲.
- 基于生成函数的处理手法, 部分经典算法, 以及它们一些 OI 之外的故事.
- 不追求困难性, 所以只会有比较简单但典型的例子.

今天讲什么?

- 组合计数明明处处都有递推, 所以几乎什么都可以讲.
- 基于生成函数的处理手法, 部分经典算法, 以及它们一些 OI 之外的故事.
- 不追求困难性, 所以只会有比较简单但典型的例子.
- 所以, 以下内容**全都不会讲**:
  - ~~UOJ593 新年的军队~~
  - ~~UOJ633 你将如闪电般归来~~
  - ~~Codeforces1687F Koishi's Unconscious Permutation~~
  - ~~SDOI2022 多边形~~
  - ~~CTS2023 另一个欧拉数问题~~

今天讲什么?

- 组合计数明明处处都有递推, 所以几乎什么都可以讲.
- 基于生成函数的处理手法, 部分经典算法, 以及它们一些 OI 之外的故事.
- 不追求困难性, 所以只会有比较简单但典型的例子.
- 所以, 以下内容**全都不会讲**:
  - ~~UOJ593 新年的军队~~
  - ~~UOJ633 你将如闪电般归来~~
  - ~~Codeforces1687F Koishi's Unconscious Permutation~~
  - ~~SDOI2022 多边形~~
  - ~~CTS2023 另一个欧拉数问题~~
- 当然, 欢迎大家补题!

1. **组合类与生成函数**
   - 组合构造的字典
   - 连通图计数
   - n 王问题
2. **有关递推式的算法**
   - 半在线卷积的更快算法 — 超越 “CDQ 分治”
   - 线性递推的 Bostan–Mori 算法
   - 多项式 Euclid 算法
   - Hermite–Padé 逼近
3. **整式递推的理论**
   - 为什么要研究整式递推
   - 线性空间的表述方式
   - 代数幂级数
   - 多元微分有限
   - 整式递推在 OI 中的未来

从组合类到生成函数

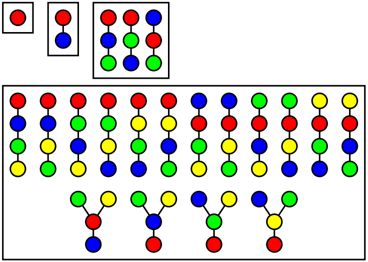


图: n 个顶点的 Cayley 树

数列

$$\{a_n\}_{n\geq 0} = \{0, 1, 1, 3, 16, \ldots\} \tag{1}$$

Generating Function
生 成 函 数

$$A(x) = \sum_{n=1}^{\infty} n^{n-2} \cdot x^n \tag{2}$$

- 组合类:

$$\mathscr{A} = \{\alpha_1, \alpha_2, \ldots, \alpha_k, \ldots\} \tag{3}$$

- 生成函数:

$$A(x) = \sum_{\alpha \in \mathscr{A}} x^{|\alpha|}. \tag{4}$$

基本运算

	$\mathscr{A}$ $= \{\alpha_1, \alpha_2, \ldots\}$	$A(x)$ $= \sum_{\alpha \in \mathscr{A}} x^{\lvert\alpha\rvert}$	a_n $= \#\{\alpha : \lvert\alpha\rvert = n\}$
无交并	$\mathscr{C} = \mathscr{A} \sqcup \mathscr{B}$	$C = A + B$	$c_n = a_n + b_n$
积	$\mathscr{C} = \mathscr{A} \times \mathscr{B}$	$C = A \cdot B$	$c_n = \sum_{k=0}^{n} a_k b_{n-k}$
序列	$\mathscr{B} = \mathrm{Seq}\, \mathscr{A}$ $= \epsilon \sqcup \mathscr{A} \times \mathscr{B}$	$B = 1 + A + A^2 + \cdots$ $= 1 + A \cdot B$ $= \frac{1}{1-A}$	$b_n = \sum_{k=1}^{n} a_k b_{n-k}$
多重集	$\mathscr{B} = \mathrm{MSet}\, \mathscr{A}$ $= \prod_{\alpha \in \mathscr{A}} (\mathrm{Seq}\, \alpha)$	$= \prod_{n=1}^{\infty} (1 - x^n)^{-a_n}$	?
幂集	$\mathscr{B} = \mathrm{Set}\, \mathscr{A}$ $= \prod_{\alpha \in \mathscr{A}} (\epsilon \sqcup \alpha)$	$= \prod_{n=1}^{\infty} (1 + x^n)^{a_n}$	?

同一个世界, 不同的梦想

- 如果 $\mathscr{A} = \mathrm{MSet}\,\mathscr{B}$, 那么

$$A(x) = \exp\left(\sum_{k=1}^{\infty} \frac{B(x^k)}{k}\right), \tag{5}$$

$$B(x) = \sum_{k=1}^{\infty} \frac{\mu(k)}{k} \log A(x^k). \tag{6}$$

- 考虑同一个生成函数

$$A(x) = \frac{1}{1-qx} = 1 + qx + q^2x^2 + \cdots, \tag{7}$$

- 对于组合类 $\mathscr{A}$ 的两种解释: 字符集为 $|\Sigma| = q$ 的字符串 Σ^*, 或有限域上的多项式 $\mathbb{F}_q[T]$.
- 组合类 $\mathscr{B}$ 的解释: 字符串的 Lyndon 分解, 或者 $\mathbb{F}_q[T]$ 分解成不可约因子之乘积.

同一个世界, 不同的梦想

- 组合类 $\mathscr{B}$ 的解释:

$$\Sigma^* = \text{MSet}[\mathscr{L}yndon] \tag{8}$$

$$\mathbb{F}_q[T] = \text{MSet}[\mathscr{I}rreducible]. \tag{9}$$

- 得到同样的生成函数和数列:

$$B(x) = \sum_{k=1}^{\infty} \frac{\mu(k)}{k} \log \frac{1}{1-qx^k} \tag{10}$$

$$b_n = \frac{1}{n} \sum_{k|n} \mu(k) q^{n/k}. \tag{11}$$

- Lyndon 串的组合意义还算好理解, 但后者的组合意义恐怕需要一点更多的知识.
- Lyndon 串和不可约多项式之间的双射也是不太显然的, 这个双射也不是很典则, 一般来说要选取域扩张 $\mathbb{F}_{q^n}/\mathbb{F}_q$ 的一个正规基.

指数生成函数的基本运算

如果 α,β 有 $\binom{|\alpha|+|\beta|}{|\alpha|}$ 种组合方式, 那么就要考虑

$$\frac{x^n}{n!}\cdot\frac{x^m}{m!}=\binom{n+m}{n}\frac{x^{n+m}}{(n+m)!}. \tag{12}$$

	$\mathscr{A}$ $=\{\alpha_1,\alpha_2,\dots\}$	$A(x)$ $=\sum_{\alpha\in\mathscr{A}} x^{\lvert\alpha\rvert}$	a_n $=\#\{\alpha:\lvert\alpha\rvert=n\}$
无交并	$\mathscr{C}=\mathscr{A}\sqcup\mathscr{B}$	$C=A+B$	$c_n=a_n+b_n$
积	$\mathscr{C}=\mathscr{A}\times\mathscr{B}$	$C=A\cdot B$	$c_n=\sum_{k=0}^n\binom{n}{k}a_kb_{n-k}$
k 元集	$\mathscr{B}=\mathrm{MSet}_k\,\mathscr{A}$	$B=A^k/k!$	?
多重集	$\mathscr{B}=\mathrm{MSet}\,\mathscr{A}$ $=\bigsqcup_{k=0}^{\infty}\mathrm{MSet}_k\,\mathscr{A}$	$B=\exp A$	?

微分算子

- 定义

$$\partial\left(\sum_{n=0}^{\infty} a_n x^n\right) = \sum_{n=0}^{\infty} a_n \cdot n x^{n-1}, \tag{13}$$

- 对于普通生成函数, 有

$$\partial \cdot x^n = n x^{n-1}. \tag{14}$$

- 对于指数型生成函数, 有关递推式的算法

$$\partial \cdot \frac{x^n}{n!} = \frac{x^{n-1}}{(n-1)!}. \tag{15}$$

- 从组合意义的角度, 它们相当于对于组合类进行了什么变换?

和微分有关的运算律

直接从组合意义的角度, 解释如下运算律:

- 可加性:
$$\partial(A+B)=\partial A+\partial B. \tag{16}$$
- Lebniz 律:
$$\partial(A\cdot B)=(\partial A)\cdot B+A\cdot(\partial B). \tag{17}$$
- 复合:
$$\partial(A\circ B)=((\partial A)\circ B)\cdot(\partial B). \tag{18}$$

多重集构造的递推式

- 微分方程 $\Longleftrightarrow$ 组合解释.

$$\mathscr{B} = \mathrm{MSet}\,\mathscr{A} \tag{19}$$

$$B = \exp A \tag{20}$$

$$B' = B \cdot A' \tag{21}$$

$$b_n = \sum_{k=1}^{n} \binom{n-1}{k-1} a_k b_{n-k}. \tag{22}$$

例子 — 连通图

设 $\mathscr{G}$ 为无向图构成的组合类, $\mathscr{C}$ 为无向连通图构成的组合类, 那么

$$\mathscr{G} = \mathrm{MSet}\,\mathscr{C}. \tag{23}$$

尝试解释以下两种不同的递推式, 分别从 **组合意义** 和 **代数推导**:

$$C_n = 2^{n(n-1)/2} - \sum_k \binom{n-1}{k-1} C_k 2^{(n-k)(n-k-1)/2}, \tag{24}$$

$$C_n = \sum_k \binom{n-2}{k-1} C_k C_{n-k} \cdot (2^k - 1). \tag{25}$$

例子 — n 王问题

- 有多少 n 阶排列 σ 使得相邻两项的差的绝对值不是 1?

例子 — n 王问题

- 有多少 n 阶排列 σ 使得相邻两项的差的绝对值不是 1?
- 查表发现这被收录于 整 数 序 列 百 科 (**E**ncyclopedia of **I**nteger **S**equences) 的第 A002464 项. 设这个数列叫做 A_n, 有如下递推式:

$$A_n = (n+1)A_{n-1} - (n-2)A_{n-2} - (n-5)A_{n-3} + (n-3)A_{n-4}. \tag{26}$$

如何证明?

组合证明? 有的, 但是...

一道组合题的线性时间做法 - 递推树上递推果，递推树下你和我

狗雷布是真的伊

真的狗啊真的狗

93 人赞同了该文章

前言：

这篇文章说的是BZOJ上的4321号题目（昨天可爱的小灰机 @FFjet 给我的），然后本龙做了一天，写了一下午，改了半个晚上，才出来这篇文章。

原题里给的 n 是有范围的， $1 \leq n \leq 1000$ ，也就是说，爆搜无望，原题目要求基本上 $O(n^3)$ 到顶了。下面解法的线性的复杂度已经尽我所能了。现在网上大部分做法是 $O(n^2)$ 的动态规划。（事实上我要是最后不化简也是 $O(n^2)$ 的，最后化简成 $O(n)$ 的算法）

好吧这个题目真的难，如果改成证明题的话，放到CMO里面也可以当3和6了qwq

现在，启动发动机，开始起飞——

https://zhuanlan.zhihu.com/p/56537011

长达几页纸的双射... 有没有更简单的方法?

生成函数与递推式

对于相邻关系容斥, 可以写出生成函数

$$\sum_n A_n x^n = \left(\sum_{n=0}^{\infty} n!T^n\right) \circ \left(x - 2x^2 + 2x^3 - 2x^4 + 2x^5 + \cdots\right) \tag{27}$$

$$= \left(\sum_{n=0}^{\infty} n!T^n\right) \circ \left(x\frac{1-x}{1+x}\right). \tag{28}$$

生成函数与递推式

对于相邻关系容斥, 可以写出生成函数

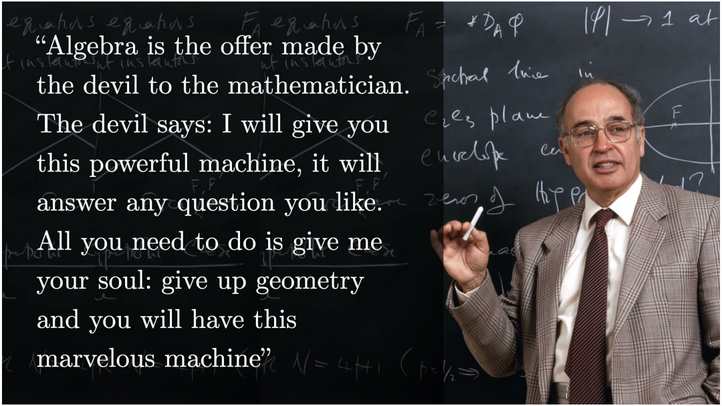
$$\sum_n A_n x^n = \left(\sum_{n=0}^{\infty} n!T^n\right) \circ \left(x - 2x^2 + 2x^3 - 2x^4 + 2x^5 + \cdots\right) \tag{27}$$

$$= \left(\sum_{n=0}^{\infty} n!T^n\right) \circ \left(x\frac{1-x}{1+x}\right). \tag{28}$$

记 $S(T) = \sum_{n=0}^{\infty} n!T^n$, 将递推式 $S_n = nS_{n-1} + [n=0]$ 转化为生成函数的微分方程

$$S(T) = 1 + (S(T)\cdot T)' \cdot T \tag{29}$$

$$S = 1 + TS + T^2S'. \tag{30}$$



— Michael Atiyah

~~生成函数是魔鬼和我们的交易. 魔鬼说: 我给你这个强大的机器,~~
~~它能回答任何你想问的问题. 但是, 你必须付出代价,~~
~~你必须给我你的灵魂: 放弃组合意义,~~
~~然后你就能得到这台威力无穷的机器!~~

课间休息, 思考题

记 $\mathscr{A}$ 为 2-正则图构成的组合类, 证明其指数型生成函数是

$$A(x) = \exp\left(-\frac{x}{2} - \frac{x^2}{4}\right) \frac{1}{\sqrt{1-x}}. \tag{31}$$

生成函数 / 多项式 — 算法

以下内容不会讲, 但是只假设它存在, 对后续内容的理解也基本没有影响.

- 快速 Fourier 变换: 高效计算 $A(x)B(x) \bmod x^n$.
- Newton 迭代法: 高效计算 $A(x)^{-1}$, $\log A(x)$, $\exp A(x)$ 等基本初等函数.

时间一般认为是 $\mathcal{O}(n \log n)$ 的, 但仔细思考计算模型会发现并不显然, 这里就记作 $\mathsf{M}(n)$.

小故事: 整数乘法的长征

- Колмогóров 的猜测: $\mathrm{M}(N) = \Omega(N^2)$

小故事: 整数乘法的长征

- Колмогóров 的猜测: $\mathrm{M}(N) = \Omega(N^2)$

分治乘法!

- $\mathrm{M}(N) = \mathcal{O}\left(N^{\log 3/\log 2}\right)$: [Karatsuba 1962]

小故事: 整数乘法的长征

- Колмогóров 的猜测: $\mathrm{M}(N) = \Omega(N^2)$

分治乘法!

- $\mathrm{M}(N) = \mathcal{O}\left(N^{\log 3/\log 2}\right)$: [Karatsuba 1962]
- $\mathrm{M}(N) = N 2^{\mathcal{O}\left(\sqrt{\log N}\right)}$: [Toom 1963], [Schönhage 1966], [Knuth 1969]

小故事: 整数乘法的长征

- Колмогóров 的猜测: $\mathrm{M}(N) = \Omega(N^2)$

分治乘法!

- $\mathrm{M}(N) = \mathscr{O}\left(N^{\log 3/\log 2}\right)$: [Karatsuba 1962]
- $\mathrm{M}(N) = N 2^{\mathscr{O}\left(\sqrt{\log N}\right)}$: [Toom 1963], [Schönhage 1966], [Knuth 1969]

快速 Fourier 变换 **[Gauß 1876]**, [Cooley–Tukey 1965], 但单位根怎么存?

- $\mathrm{M}(N) = \mathscr{O}(N \log N \log\log N \log\log\log N \cdots)$: [Pollard 1971]
- $\mathrm{M}(N) = \mathscr{O}(N \log N \log\log N)$: **[Schönhage–Strassen 1971]** GMP «Arithmetic without limitations»

小故事: 整数乘法的长征

- Колмогóров 的猜测: $\mathrm{M}(N) = \Omega(N^2)$

分治乘法!

- $\mathrm{M}(N) = \mathcal{O}\left(N^{\log 3/\log 2}\right)$: [Karatsuba 1962]
- $\mathrm{M}(N) = N2^{\mathcal{O}\left(\sqrt{\log N}\right)}$: [Toom 1963], [Schönhage 1966], [Knuth 1969]

快速 Fourier 变换 **[Gauß 1876]**, [Cooley–Tukey 1965], 但单位根怎么存?

- $\mathrm{M}(N) = \mathcal{O}(N \log N \log\log N \log\log\log N \cdots)$: [Pollard 1971]
- $\mathrm{M}(N) = \mathcal{O}(N \log N \log\log N)$: **[Schönhage–Strassen 1971]** GMP «Arithmetic without limitations»
- $\mathrm{M}(N) = N \log N 2^{\mathcal{O}(\log^* N)}$: [Fürer 2007], Harvey, van der Hoeven, Lecerf...
- $\mathrm{M}(N) = \mathcal{O}(N \log N)$: [Harvey–van der Hoeven 2019]

模型是多带 Turing 机, 可以大概理解成要衡量一个一个 bit 的操作.

小故事: 整数乘法的长征

- Колмогóров 的猜测: $\mathrm{M}(N) = \Omega(N^2)$

分治乘法!

- $\mathrm{M}(N) = \mathcal{O}\left(N^{\log 3/\log 2}\right)$: [Karatsuba 1962]
- $\mathrm{M}(N) = N2^{\mathcal{O}\left(\sqrt{\log N}\right)}$: [Toom 1963], [Schönhage 1966], [Knuth 1969]

快速 Fourier 变换 **[Gauß 1876]**, [Cooley–Tukey 1965], 但单位根怎么存?

- $\mathrm{M}(N) = \mathcal{O}(N \log N \log\log N \log\log\log N \cdots)$: [Pollard 1971]
- $\mathrm{M}(N) = \mathcal{O}(N \log N \log\log N)$: **[Schönhage–Strassen 1971]** GMP
- $\mathrm{M}(N) = N \log N 2^{\mathcal{O}(\log^* N)}$: [Fürer 2007], Harvey, van der Hoeven, Lecerf...
- $\mathrm{M}(N) = \mathcal{O}(N \log N)$: [Harvey–van der Hoeven 2019]

模型是多带 Turing 机, 可以大概理解成要衡量一个一个 bit 的操作.

- 如果 $\overset{\text{Network Coding Conjecture}}{\text{网络编码猜想}}$ 成立, 那么这是不可改进的.
  [Afshani–Freksen–Kamma–Larsen 2019]

半在线卷积

- 一般的卷积式

$$c_n = \sum_k a_k b_{n-k} \tag{32}$$

也可以看做是一个递推式, 如果我们只有知道了 c_n 才知道 a_{n+1}, b_{n+1}. 这是 在线卷积 (Relaxed Convolution) 问题.

半在线卷积

- 一般的卷积式

$$c_n = \sum_k a_k b_{n-k} \tag{32}$$

也可以看做是一个递推式, 如果我们只有知道了 c_n 才知道 a_{n+1}, b_{n+1}. 这是 在线卷积 (Relaxed Convolution) 问题.

- 如果序列 b 是一开始就完全知道的, 这是 **半** 在线卷积 (**Semi** Relaxed Convolution) 问题.

半在线卷积

- 一般的卷积式

$$c_n = \sum_k a_k b_{n-k} \tag{32}$$

也可以看做是一个递推式, 如果我们只有知道了 c_n 才知道 a_{n+1}, b_{n+1}. 这是 在线卷积 (Relaxed Convolution) 问题.

- 如果序列 b 是一开始就完全知道的, 这是 半在线卷积 (**Semi** Relaxed Convolution) 问题.
- 通过外层分治可以发现, 在线卷积并不比半在线卷积要难.

半在线卷积的算法 [van der Hoeven 2002, 2007]

- 直接的半在线卷积算法: 每次分治成两半, 递归求解, 时间复杂度 $\mathscr{O}(N\log^2 N)$.

半在线卷积的算法 [van der Hoeven 2002, 2007]

- 直接的半在线卷积算法: 每次分治成两半, 递归求解, 时间复杂度 $\mathscr{O}(N\log^2 N)$.
- 每次分成 $B=\mathscr{O}(\log N)$ 块: 时间复杂度 $\mathscr{O}\left(\frac{N\log^2 N}{\log\log N}\right)$.

半在线卷积的算法 [van der Hoeven 2002, 2007]

- 直接的半在线卷积算法: 每次分治成两半, 递归求解, 时间复杂度 $\mathscr{O}(N\log^2 N)$.
- 每次分成 $B=\mathscr{O}(\log N)$ 块: 时间复杂度 $\mathscr{O}\left(\frac{N\log^2 N}{\log\log N}\right)$.
- 形如 $T(N)=3\sqrt{N}T(\sqrt{N})+\mathscr{O}(N\log N)$ 的递归式:

$$\mathscr{O}\left(N(\log N)^{\log 3/\log 2}\right) \tag{33}$$

半在线卷积的算法 [van der Hoeven 2002, 2007]

- 直接的半在线卷积算法: 每次分治成两半, 递归求解, 时间复杂度 $\mathscr{O}(N\log^2 N)$.
- 每次分成 $B=\mathscr{O}(\log N)$ 块: 时间复杂度 $\mathscr{O}\left(\frac{N\log^2 N}{\log\log N}\right)$.
- 形如 $T(N)=3\sqrt{N}T(\sqrt{N})+\mathscr{O}(N\log N)$ 的递归式:

$$\mathscr{O}\left(N(\log N)^{\log 3/\log 2}\right) \tag{33}$$

- 形如 $T(N)=2\ell N^{1-1/\ell}T(N^{1/\ell})+\mathscr{O}(\ell N\log N)$ 的递归式:

$$\mathscr{O}\left(N\log N\exp\left(2\sqrt{\log 2\log\log N}\right)\right) \tag{34}$$

半在线卷积的算法 [van der Hoeven 2002, 2007]

- 直接的半在线卷积算法: 每次分治成两半, 递归求解, 时间复杂度 $\mathscr{O}(N\log^2 N)$.
- 每次分成 $B=\mathscr{O}(\log N)$ 块: 时间复杂度 $\mathscr{O}\left(\frac{N\log^2 N}{\log\log N}\right)$.
- 形如 $T(N)=3\sqrt{N}T(\sqrt{N})+\mathscr{O}(N\log N)$ 的递归式:

$$\mathscr{O}\left(N(\log N)^{\log 3/\log 2}\right) \tag{33}$$

- 形如 $T(N)=2\ell N^{1-1/\ell}T(N^{1/\ell})+\mathscr{O}(\ell N\log N)$ 的递归式:

$$\mathscr{O}\left(N\log N\exp\left(2\sqrt{\log 2\log\log N}\right)\right) \tag{34}$$

- 没有平衡?

$$\mathrm{R}(n)=\mathscr{O}\left(N\log N\exp\left(\sqrt{2\log 2\log\log N}\right)\sqrt{\log\log N}\right) \tag{35}$$

线性递推的简洁算法 [Bostan–Mori 2021]

1. 将生成函数写作 $P(x)/Q(x)$ 的形式

线性递推的简洁算法 [Bostan–Mori 2021]

1. 将生成函数写作 $P(x)/Q(x)$ 的形式
2. 不妨分子分母同乘 $Q(-x)$, 得到 $P(x)Q(-x)/Q(x)Q(-x)$, 分母有什么特点?

线性递推的简洁算法 [Bostan–Mori 2021]

1. 将生成函数写作 $P(x)/Q(x)$ 的形式
2. 不妨分子分母同乘 $Q(-x)$, 得到 $P(x)Q(-x)/Q(x)Q(-x)$, 分母有什么特点?
3. 求第 K 项的时间复杂度: $\mathcal{O}(\mathsf{M}(N)\log K)$. **只需要实现多项式乘法.**

多项式 Euclid

- 给定多项式 $A(T),B(T)$, 求 $X(T),Y(T)$ 使得

$$AX+BY=\gcd(A,B). \tag{36}$$

多项式 Euclid

- 给定多项式 $A(T),B(T)$, 求 $X(T),Y(T)$ 使得

$$AX+BY=\gcd(A,B). \tag{36}$$

- 说真的, 我们除了辗转相除法以外没有别的什么思路.

$$A_0,B_0 \tag{37}$$
$$A_1=B_0,B_1=A_0 \bmod B_0 \tag{38}$$
$$A_2=B_1,B_2=A_1 \bmod B_1 \tag{39}$$
$$\cdots \tag{40}$$
$$A_\ell=B_{\ell-1},B_\ell=A_{\ell-1} \bmod B_{\ell-1} \tag{41}$$

不妨设 $\deg A > \deg B$.

$$A_0, B_0 \tag{42}$$

$$A_1 = B_0, B_1 = A_0 - B_0 \cdot Q_1 \tag{43}$$

$$A_2 = B_1, B_2 = A_1 - B_1 \cdot Q_2 \tag{44}$$

$$\ldots \tag{45}$$

$$A_\ell = B_{\ell-1}, B_\ell = A_{\ell-1} - B_{\ell-1} \cdot Q_\ell, \tag{46}$$

不妨设 $\deg A > \deg B$.

$$A_0, B_0 \tag{42}$$
$$A_1 = B_0, B_1 = A_0 - B_0 \cdot Q_1 \tag{43}$$
$$A_2 = B_1, B_2 = A_1 - B_1 \cdot Q_2 \tag{44}$$
$$\ldots \tag{45}$$
$$A_\ell = B_{\ell-1}, B_\ell = A_{\ell-1} - B_{\ell-1} \cdot Q_\ell, \tag{46}$$

- 如果有一个函数 $\mathrm{HalfGCD}_N(A,B)$ 将两个次数 $< N$ 的多项式求出 $Q_1, \ldots, Q_\ell$ 使得 $\deg B_\ell < N/2$, 但 $\deg A_\ell \geq N/2$...

不妨设 $\deg A > \deg B$.

$$A_0, B_0 \tag{42}$$

$$A_1 = B_0, B_1 = A_0 - B_0 \cdot Q_1 \tag{43}$$

$$A_2 = B_1, B_2 = A_1 - B_1 \cdot Q_2 \tag{44}$$

$$\cdots \tag{45}$$

$$A_\ell = B_{\ell-1}, B_\ell = A_{\ell-1} - B_{\ell-1} \cdot Q_\ell, \tag{46}$$

- 如果有一个函数 $\text{HalfGCD}_N(A,B)$ 将两个次数 $< N$ 的多项式求出 $Q_1, \ldots, Q_\ell$ 使得 $\deg B_\ell < N/2$, 但 $\deg A_\ell \geq N/2$...
- 那么调用 $\log N$ 次就可以得到完整的 Euclid 过程商的序列, 而且复杂度可以被主定理控制.

折半 Euclid 算法 [Knuth 1970] [Schönhage 1971] [Moenck 1973]

不妨设 N 是 2 的幂, $\deg A > \deg B$.

- 函数 $\mathrm{HalfGCD}_N(A,B)$ 的目的: 将两个次数 $< N$ 的多项式求出 $Q_1,\ldots,Q_\ell$ 使得 $\deg B_\ell < N/2$, 但 $\deg A_\ell \geq N/2$.

折半 Euclid 算法 [Knuth 1970] [Schönhage 1971] [Moenck 1973]

不妨设 N 是 2 的幂, $\deg A > \deg B$.

- 函数 $\mathrm{HalfGCD}_N(A,B)$ 的目的: 将两个次数 $< N$ 的多项式求出 $Q_1,\ldots,Q_\ell$ 使得 $\deg B_\ell < N/2$, 但 $\deg A_\ell \geq N/2$.
- **前期的计算不会影响太低位:** $\deg(Q_1\cdots Q_\ell) = \deg A_0 - \deg A_\ell < N/2$.

折半 Euclid 算法 [Knuth 1970] [Schönhage 1971] [Moenck 1973]

不妨设 N 是 2 的幂, $\deg A > \deg B$.

- 函数 $\mathrm{HalfGCD}_N(A,B)$ 的目的: 将两个次数 $< N$ 的多项式求出 $Q_1,\ldots,Q_\ell$ 使得 $\deg B_\ell < N/2$, 但 $\deg A_\ell \geq N/2$.
- **前期的计算不会影响太低位:** $\deg(Q_1\cdots Q_\ell) = \deg A_0 - \deg A_\ell < N/2$.
- 如果对于 $A = T^L A' + O(T^{L-1})$, $B = T^L B' + O(T^{L-1})$ 做 $\mathrm{HalfGCD}_{N/2}(A',B')$...

折半 Euclid 算法 [Knuth 1970] [Schönhage 1971] [Moenck 1973]

不妨设 N 是 2 的幂, $\deg A > \deg B$.

- 函数 $\mathrm{HalfGCD}_N(A,B)$ 的目的: 将两个次数 $< N$ 的多项式求出 $Q_1,\ldots,Q_\ell$ 使得 $\deg B_\ell < N/2$, 但 $\deg A_\ell \geq N/2$.
- **前期的计算不会影响太低位:** $\deg(Q_1 \cdots Q_\ell) = \deg A_0 - \deg A_\ell < N/2$.
- 如果对于 $A = T^L A' + O(T^{L-1})$, $B = T^L B' + O(T^{L-1})$ 做 $\mathrm{HalfGCD}_{N/2}(A',B')$...
- 由于我们可以写成矩阵,

$$A_i = B_{i-1}, B_i = A_{i-1} - B_{i-1} \cdot Q_i \iff \begin{pmatrix} A_i \\ B_i \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & -Q_i \end{pmatrix} \begin{pmatrix} A_{i-1} \\ B_{i-1} \end{pmatrix}. \tag{47}$$

- 有

$$\begin{pmatrix} 0 & 1 \\ 1 & -Q_\ell \end{pmatrix} \cdots \begin{pmatrix} 0 & 1 \\ 1 & -Q_1 \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} = T^L \begin{pmatrix} A' \\ B' \end{pmatrix} + O(T^{L-1}) \cdot O(T^{N/4-1}). \tag{48}$$

折半 Euclid 算法 [Knuth 1970] [Schönhage 1971] [Moenck 1973]

- 如果对于 $A = T^L A' + O(T^{L-1})$, $B = T^L B' + O(T^{L-1})$ 做 $\mathrm{HalfGCD}_{N/2}(A', B')$, 有

$$\begin{pmatrix} 0 & 1 \\ 1 & -Q_\ell \end{pmatrix} \cdots \begin{pmatrix} 0 & 1 \\ 1 & -Q_1 \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} = T^L \begin{pmatrix} A' \\ B' \end{pmatrix} + O(T^{L-1}) \cdot O(T^{N/4-1}). \tag{49}$$

折半 Euclid 算法 [Knuth 1970] [Schönhage 1971] [Moenck 1973]

- 如果对于 $A = T^L A' + O(T^{L-1})$, $B = T^L B' + O(T^{L-1})$ 做 $\mathrm{HalfGCD}_{N/2}(A', B')$, 有

$$\begin{pmatrix} 0 & 1 \\ 1 & -Q_\ell \end{pmatrix} \cdots \begin{pmatrix} 0 & 1 \\ 1 & -Q_1 \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} = T^L \begin{pmatrix} A' \\ B' \end{pmatrix} + O(T^{L-1}) \cdot O(T^{N/4-1}). \tag{49}$$

- 第一次取 $L = N/2$, 将 (A, B) 约化到 $\deg A \geq 3/4 N > \deg B$, 然后做一次取模.

折半 Euclid 算法 [Knuth 1970] [Schönhage 1971] [Moenck 1973]

- 如果对于 $A = T^L A' + O(T^{L-1})$, $B = T^L B' + O(T^{L-1})$ 做 $\mathrm{HalfGCD}_{N/2}(A',B')$, 有

$$\begin{pmatrix} 0 & 1 \\ 1 & -Q_\ell \end{pmatrix} \cdots \begin{pmatrix} 0 & 1 \\ 1 & -Q_1 \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} = T^L \begin{pmatrix} A' \\ B' \end{pmatrix} + O(T^{L-1}) \cdot O(T^{N/4-1}). \tag{49}$$

- 第一次取 $L = N/2$, 将 (A,B) 约化到 $\deg A \geq 3/4N > \deg B$, 然后做一次取模.
- 第二次取 $L = N/4$, 将 (A,B) 约化到 $\deg A \geq 1/2N > \deg B$.

折半 Euclid 算法 [Knuth 1970] [Schönhage 1971] [Moenck 1973]

- 如果对于 $A = T^L A' + O(T^{L-1})$, $B = T^L B' + O(T^{L-1})$ 做 $\mathrm{HalfGCD}_{N/2}(A', B')$, 有

$$\begin{pmatrix} 0 & 1 \\ 1 & -Q_\ell \end{pmatrix} \cdots \begin{pmatrix} 0 & 1 \\ 1 & -Q_1 \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} = T^L \begin{pmatrix} A' \\ B' \end{pmatrix} + O(T^{L-1}) \cdot O(T^{N/4-1}). \tag{49}$$

- 第一次取 $L = N/2$, 将 (A, B) 约化到 $\deg A \geq 3/4 N > \deg B$, 然后做一次取模.
- 第二次取 $L = N/4$, 将 (A, B) 约化到 $\deg A \geq 1/2 N > \deg B$.
- 这总共调用了两次分治, 有

$$T(N) = 2T(N/2) + \mathcal{O}(\mathsf{M}(N)), \tag{50}$$

解得 $T(N) = \mathcal{O}(\mathsf{M}(N) \log N)$.

连分式展开

我们求出的序列的一种直观解释:

$$\frac{A}{B} = Q_1 + \frac{A \bmod B}{B} \tag{51}$$

$$= Q_1 + \frac{1}{B/(A \bmod B)} \tag{52}$$

$$= Q_1 + \cfrac{1}{Q_2 + \cfrac{1}{Q_3 + \ddots}} \tag{53}$$

$$:= [Q_1; Q_2, \ldots, Q_\ell]. \tag{54}$$

线性递推式重建

- 已知一个 $\leq N$ 阶线性递推式的前 $2N$ 项 $a_0, \ldots, a_{2N-1}$, 求递推式.

线性递推式重建

- 已知一个 $\le N$ 阶线性递推式的前 $2N$ 项 $a_0, \dots, a_{2N-1}$, 求递推式.
- 我们知道, 这是要找 P, Q 满足 $A \equiv P/Q \pmod{x^{2N}}$, 且 $\deg P < N, \deg Q \le N$.

线性递推式重建

- 已知一个 $\le N$ 阶线性递推式的前 $2N$ 项 $a_0,\dots,a_{2N-1}$, 求递推式.
- 我们知道, 这是要找 P,Q 满足 $A \equiv P/Q \pmod{x^{2N}}$, 且 $\deg P < N, \deg Q \le N$.

定义 (Padé 逼近)

给定 $A(x)$, 求 $\deg P \le N_1$, $\deg Q \le N_2$ 满足

$$P - AQ \equiv 0 \pmod{x^{N_1+N_2+1}}. \tag{55}$$

线性递推式重建

- 已知一个 $\le N$ 阶线性递推式的前 $2N$ 项 $a_0,\dots,a_{2N-1}$, 求递推式.
- 我们知道, 这是要找 P,Q 满足 $A \equiv P/Q \pmod{x^{2N}}$, 且 $\deg P < N, \deg Q \le N$.

定义 (Padé 逼近)

给定 $A(x)$, 求 $\deg P \le N_1$, $\deg Q \le N_2$ 满足

$$P - AQ \equiv 0 \pmod{x^{N_1+N_2+1}}. \tag{55}$$

定义 (有理函数重建)

给定 $A(x)$ 和模 $M(x)$, 求 $\deg P \le N_1$, $\deg Q \le N_2$ 满足

$$P - AQ \equiv 0 \pmod{M}, \tag{56}$$

其中 $\deg M = N_1 + N_2 + 1$.

连分式展开 $\Longrightarrow$ 有理函数重建

定义 (有理函数重建)

给定 $A(x)$ 和模 $M(x)$, 求 $\deg P \le N_1$, $\deg Q \le N_2$ 满足

$$P - AQ \equiv 0 \pmod{M}, \tag{57}$$

其中 $\deg M = N_1 + N_2 + 1$.

连分式展开 $\Longrightarrow$ 有理函数重建

定义 (有理函数重建)

给定 $A(x)$ 和模 $M(x)$, 求 $\deg P \le N_1$, $\deg Q \le N_2$ 满足

$$P = AQ + BM, \tag{57}$$

其中 $\deg M = N_1 + N_2 + 1$.

连分式展开 $\Longrightarrow$ 有理函数重建

定义 (有理函数重建)

给定 $A(x)$ 和模 $M(x)$, 求 $\deg P \le N_1$, $\deg Q \le N_2$ 满足

$$P = AQ + BM, \tag{57}$$

其中 $\deg M = N_1 + N_2 + 1$.

$$A_0 = M \tag{58}$$

$$A_1 = A \tag{59}$$

$$A_2 = A_0 - A_1 \cdot Q_1 \tag{60}$$

$$\dots \tag{61}$$

连分式展开 $\Longrightarrow$ 有理函数重建

定义 (有理函数重建)

给定 $A(x)$ 和模 $M(x)$, 求 $\deg P \le N_1$, $\deg Q \le N_2$ 满足

$$P = AQ + BM, \tag{57}$$

其中 $\deg M = N_1 + N_2 + 1$.

$$A_0 = M \tag{58}$$

$$\overbrace{A_1}^{\deg < \deg A_0} = A \qquad = O(x^{\deg A_0 - \deg A_0}) \cdot (M, A) \tag{59}$$

$$\overbrace{A_2}^{\deg < \deg A_1} = A_0 - A_1 \cdot Q_1 \qquad = O(x^{\deg A_0 - \deg A_1}) \cdot (M, A) \tag{60}$$

$$\ldots \tag{61}$$

间奏: 纠错码的快速算法

纠错码是通信的基础, 考虑固定的一个有限的字符集 Σ, 一个**纠错码**可以看做一个函数 $C : \Sigma^n \to \Sigma^m$ $(m \geq n)$.
所以为了保证纠错码的可靠性, 我们关心 "距离":

$$d = \min_{\substack{x,y \in \Sigma^n \\ x \neq y}} \delta(C(x), C(y)), \tag{62}$$

其中 δ 是 Σ^m 上的 Hamming 距离. 易见, 如果传输中错误的字符数量 $< d/2$, 那么我们可以完美地恢复原来的信息.

基于多项式求值的编码 [Reed–Solomon 1960]

令字符集为有限域 $\mathbb{F}$ 满足 $|\mathbb{F}| \geq m$, 令 $\alpha_1, \ldots, \alpha_m$ 是 $\mathbb{F}$ 的 m 个不同的元素, 考虑如下的映射:

$$C(a_0, \ldots, a_{n-1}) \mapsto \left(\sum_{j=0}^{n-1} a_j \alpha_i^j \right)_{i=1}^{m}, \tag{63}$$

也即将信息 $a_0, \ldots, a_{n-1}$ 看做一个 $n-1$ 次多项式 $f(x) = \sum_{j=0}^{n-1} a_j x^j$, 在 m 个给定点处的取值.

注意到 n 个点值就足够确定一个 $n-1$ 次多项式, 所以 **Reed–Solomon 码**的距离满足 $d \geq m - n + 1$.

Reed–Solomon 编码的快速纠错 [Berlekamp–Welch 1986]

给一个 $n-1$ 次多项式 $f(x)$, 如果 $f(\alpha_1),\dots,f(\alpha_m)$ 中有 $\le (m-n)/2$ 个错误, 一定可以唯一地还原出 f 的系数, 但是如何快速计算?

不妨设给定的 $y_1,\dots,y_m$ 确定出来的多项式是 $g(x)$, 那么 $g(x)-f(x)$ 只在 $\beta_1,\dots,\beta_k$ 上非零, 所以

$$(g(x)-f(x))(x-\beta_1)\cdots(x-\beta_k)\equiv 0 \pmod{(x-\alpha_1)\cdots(x-\alpha_m)}, \tag{64}$$

注意写作 $r(x)=(x-\beta_1)\cdots(x-\beta_k)$, 那么

$$\deg r+\deg(f\cdot r)=k+(n-1+k)\le m-1, \tag{65}$$

可以考虑直接对 $g(x)$ 进行有理函数重建.

更加一般的问题

定义 (Hermite–Padé 逼近)

给定多项式 $A_1,\dots,A_m$ 和度数限制 $d_1,\dots,d_m$, 求 $P_1,\dots,P_m$ 使得

- $\deg P_i < d_i$,
- $P_1A_1 + \cdots + P_mA_m \equiv 0 \pmod{x^{d_1+\cdots+d_m-1}}$.

更加一般的问题

定义 (Hermite–Padé 逼近)

给定多项式 $A_1, \ldots, A_m$ 和度数限制 $d_1, \ldots, d_m$, 求 $P_1, \ldots, P_m$ 使得

- $\deg P_i < d_i$,
- $P_1 A_1 + \cdots + P_m A_m \equiv 0 \pmod{x^{d_1 + \cdots + d_m - 1}}$.

Min25 BM

- 找寻整式递推式 可以直接对 $(A, A', \ldots, A^{(m-1)})$ 调用 Hermite–Padé 逼近的算法.
- 找寻最小多项式可以直接对 $(1, A, \ldots, A^{m-1})$ 调用 Hermite–Padé 逼近的算法.

更加一般的问题

定义 (Hermite–Padé 逼近)

给定多项式 $A_1,\ldots,A_m$ 和度数限制 $d_1,\ldots,d_m$, 求 $P_1,\ldots,P_m$ 使得

- $\deg P_i < d_i$,
- $P_1A_1+\cdots+P_mA_m \equiv 0 \pmod{x^{d_1+\cdots+d_m-1}}$.

Min25 BM

- 找寻整式递推式 可以直接对 $(A, A', \ldots, A^{(m-1)})$ 调用 Hermite–Padé 逼近的算法.
- 找寻最小多项式可以直接对 $(1, A, \ldots, A^{m-1})$ 调用 Hermite–Padé 逼近的算法.
- 记 $\sigma = d_1+\cdots+d_m-1$, Hermite–Padé 逼近的时间复杂度可以在 $\widetilde{\mathscr{O}}(m^{\omega-1}\sigma)$ 的时间内完成, 其中 ω 是矩阵乘法的指数*. [Labahn–Zhou 2012]

*现在 $\omega \le 2.371552$. [Williams–Xu–Xu–Zhou 2023]

解 Toeplitz 方程

定义 (Toeplitz 矩阵)

形如 $(a_{i-j})_{i,j}$ 的矩阵, 其中 a 是下标从 $-(N-1)$ 到 $N-1$ 的数列.

解 Toeplitz 方程

定义 (Toeplitz 矩阵)

形如 $(a_{i-j})_{i,j}$ 的矩阵, 其中 a 是下标从 $-(N-1)$ 到 $N-1$ 的数列.

- Toeplitz 方程可以写作

$$a(T)\cdot x(T) = b(T) + \Omega(T^{N-2}) + O(T^{2N-1}), \tag{66}$$

解 Toeplitz 方程

定义 (Toeplitz 矩阵)

形如 $(a_{i-j})_{i,j}$ 的矩阵, 其中 a 是下标从 $-(N-1)$ 到 $N-1$ 的数列.

- Toeplitz 方程可以写作

$$a(T)\cdot x(T) = b(T) + \Omega(T^{N-2}) + O(T^{2N-1}), \tag{66}$$

- 转化成

$$a(T)\cdot \underbrace{x(T)}_{\deg<N} - b(T)\cdot \underbrace{1}_{\deg<1} - 1\cdot \underbrace{r(T)}_{\deg<N-1} \equiv 0 \pmod{T^{2N-1}}. \tag{67}$$

解 Toeplitz 方程

定义 (Toeplitz 矩阵)

形如 $(a_{i-j})_{i,j}$ 的矩阵, 其中 a 是下标从 $-(N-1)$ 到 $N-1$ 的数列.

- Toeplitz 方程可以写作

$$a(T)\cdot x(T) = b(T) + \Omega(T^{N-2}) + O(T^{2N-1}), \tag{66}$$

- 转化成

$$a(T)\cdot \underbrace{x(T)}_{\deg<N} - b(T)\cdot \underbrace{1}_{\deg<1} - 1\cdot \underbrace{r(T)}_{\deg<N-1} \equiv 0 \pmod{T^{2N-1}}. \tag{67}$$

- 刚好 $N+1+(N-1)-1 = 2N-1$, 符合 Hermite–Padé 逼近的形式.

解的多项式基

- 给定 $\boldsymbol{A} \in \mathbb{F}[\![x]\!]^m$, 定义 $V_s = \{\boldsymbol{P} \in \mathbb{F}[x]^m : \boldsymbol{P} \cdot \boldsymbol{A} \equiv 0 \pmod{x^s}\}$.
- 记 $\deg(\boldsymbol{P}) = \max_{1 \le i \le m}\{\deg(\boldsymbol{P}_i)\}$, $\operatorname{type}(\boldsymbol{P})$ 为取到最大值的最大的 i.
- V_s 的 **极小基**: 对每个 i, 选取 $\boldsymbol{Q}_i$ 是 $\operatorname{type}(\boldsymbol{P}) = i$ 中度数最小的一个.

性质:

解的多项式基

- 给定 $\boldsymbol{A} \in \mathbb{F}[\![x]\!]^m$, 定义 $V_s = \{\boldsymbol{P} \in \mathbb{F}[x]^m : \boldsymbol{P} \cdot \boldsymbol{A} \equiv 0 \pmod{x^s}\}$.
- 记 $\deg(\boldsymbol{P}) = \max_{1 \le i \le m}\{\deg(\boldsymbol{P}_i)\}$, $\operatorname{type}(\boldsymbol{P})$ 为取到最大值的最大的 i.
- V_s 的 **极小基**: 对每个 i, 选取 $\boldsymbol{Q}_i$ 是 $\operatorname{type}(\boldsymbol{P}) = i$ 中度数最小的一个.

性质:

- 确实是基: $\boldsymbol{Q}_1, \ldots, \boldsymbol{Q}_m$ 可以组合出 V_s 中的元素.

解的多项式基

- 给定 $\boldsymbol{A} \in \mathbb{F}[\![x]\!]^m$, 定义 $V_s = \{\boldsymbol{P} \in \mathbb{F}[x]^m : \boldsymbol{P} \cdot \boldsymbol{A} \equiv 0 \pmod{x^s}\}$.
- 记 $\deg(\boldsymbol{P}) = \max_{1 \le i \le m}\{\deg(\boldsymbol{P}_i)\}$, $\mathrm{type}(\boldsymbol{P})$ 为取到最大值的最大的 i.
- V_s 的 **极小基**: 对每个 i, 选取 $\boldsymbol{Q}_i$ 是 $\mathrm{type}(\boldsymbol{P}) = i$ 中度数最小的一个.

性质:

- 确实是基: $\boldsymbol{Q}_1, \ldots, \boldsymbol{Q}_m$ 可以组合出 V_s 中的元素.
- 最优性: V_{md-1} 的一组极小基中, 次数最小的 $\boldsymbol{Q}_i$ 是 $\deg < d$ 的 Hermite–Padé 逼近的形式的一组解.

极小基的维护

- 按照 s 从小到大的顺序逐渐维护 V_s 的一组极小基.
- V_0 是平凡情况, 有 $\boldsymbol{Q}_{ij} = [i = j]$.
- 从 V_s 推到 V_{s+1} 的极小基 $\{\widetilde{\boldsymbol{Q}}_i\}$:

极小基的维护

- 按照 s 从小到大的顺序逐渐维护 V_s 的一组极小基.
- V_0 是平凡情况, 有 $\boldsymbol{Q}_{ij} = [i = j]$.
- 从 V_s 推到 V_{s+1} 的极小基 $\{\widetilde{\boldsymbol{Q}}_i\}$:
  - 如果 $x^{s+1} \mid \boldsymbol{Q}_i \cdot \boldsymbol{A}$, 可以直接保留, $\widetilde{\boldsymbol{Q}}_i = \boldsymbol{Q}_i$.

极小基的维护

- 按照 s 从小到大的顺序逐渐维护 V_s 的一组极小基.
- V_0 是平凡情况, 有 $\boldsymbol{Q}_{ij} = [i = j]$.
- 从 V_s 推到 V_{s+1} 的极小基 $\{\widetilde{\boldsymbol{Q}}_i\}$:
  - 如果 $x^{s+1} \mid \boldsymbol{Q}_i \cdot \boldsymbol{A}$, 可以直接保留, $\widetilde{\boldsymbol{Q}}_i = \boldsymbol{Q}_i$.
  - 否则存在 x^s 次项, 设 ℓ 是这种情况的 i 中按照 $(\deg, \mathrm{type})$ 的字典序比较下最小的那个.

极小基的维护

- 按照 s 从小到大的顺序逐渐维护 V_s 的一组极小基.
- V_0 是平凡情况, 有 $\boldsymbol{Q}_{ij}=[i=j]$.
- 从 V_s 推到 V_{s+1} 的极小基 $\{\widetilde{\boldsymbol{Q}}_i\}$:
  - 如果 $x^{s+1}\mid \boldsymbol{Q}_i\cdot\boldsymbol{A}$, 可以直接保留, $\widetilde{\boldsymbol{Q}}_i=\boldsymbol{Q}_i$.
  - 否则存在 x^s 次项, 设 ℓ 是这种情况的 i 中按照 $(\deg,\mathrm{type})$ 的字典序比较下最小的那个.
  - 对于这样的 $i\neq\ell$, 通过 $\widetilde{\boldsymbol{Q}}_i=\boldsymbol{Q}_i-\lambda\cdot\boldsymbol{Q}_\ell$ 消去 x^s 次项 (为什么 $\mathrm{type}(\widetilde{\boldsymbol{Q}}_i)=i$?)

极小基的维护

- 按照 s 从小到大的顺序逐渐维护 V_s 的一组极小基.
- V_0 是平凡情况, 有 $\boldsymbol{Q}_{ij} = [i = j]$.
- 从 V_s 推到 V_{s+1} 的极小基 $\{\widetilde{\boldsymbol{Q}}_i\}$:
  - 如果 $x^{s+1} \mid \boldsymbol{Q}_i \cdot \boldsymbol{A}$, 可以直接保留, $\widetilde{\boldsymbol{Q}}_i = \boldsymbol{Q}_i$.
  - 否则存在 x^s 次项, 设 ℓ 是这种情况的 i 中按照 $(\deg, \mathrm{type})$ 的字典序比较下最小的那个.
  - 对于这样的 $i \neq \ell$, 通过 $\widetilde{\boldsymbol{Q}}_i = \boldsymbol{Q}_i - \lambda \cdot \boldsymbol{Q}_\ell$ 消去 x^s 次项 (为什么 $\mathrm{type}(\widetilde{\boldsymbol{Q}}_i) = i$?)
  - 对于 ℓ, 必须有 $\widetilde{\boldsymbol{Q}}_\ell = x \cdot \boldsymbol{Q}_\ell$. (总得牺牲一个)

极小基的维护

- 按照 s 从小到大的顺序逐渐维护 V_s 的一组极小基.
- V_0 是平凡情况, 有 $\boldsymbol{Q}_{ij} = [i = j]$.
- 从 V_s 推到 V_{s+1} 的极小基 $\{\widetilde{\boldsymbol{Q}}_i\}$:
  - 如果 $x^{s+1} \mid \boldsymbol{Q}_i \cdot \boldsymbol{A}$, 可以直接保留, $\widetilde{\boldsymbol{Q}}_i = \boldsymbol{Q}_i$.
  - 否则存在 x^s 次项, 设 ℓ 是这种情况的 i 中按照 $(\deg, \mathrm{type})$ 的字典序比较下最小的那个.
  - 对于这样的 $i \neq \ell$, 通过 $\widetilde{\boldsymbol{Q}}_i = \boldsymbol{Q}_i - \lambda \cdot \boldsymbol{Q}_\ell$ 消去 x^s 次项 (为什么 $\mathrm{type}(\widetilde{\boldsymbol{Q}}_i) = i$?)
  - 对于 ℓ, 必须有 $\widetilde{\boldsymbol{Q}}_\ell = x \cdot \boldsymbol{Q}_\ell$. (总得牺牲一个)
- 正确性: 考虑消元.

极小基的维护

- 按照 s 从小到大的顺序逐渐维护 V_s 的一组极小基.
- V_0 是平凡情况, 有 $\boldsymbol{Q}_{ij} = [i = j]$.
- 从 V_s 推到 V_{s+1} 的极小基 $\{\widetilde{\boldsymbol{Q}}_i\}$:
  - 如果 $x^{s+1} \mid \boldsymbol{Q}_i \cdot \boldsymbol{A}$, 可以直接保留, $\widetilde{\boldsymbol{Q}}_i = \boldsymbol{Q}_i$.
  - 否则存在 x^s 次项, 设 ℓ 是这种情况的 i 中按照 $(\deg, \mathrm{type})$ 的字典序比较下最小的那个.
  - 对于这样的 $i \neq \ell$, 通过 $\widetilde{\boldsymbol{Q}}_i = \boldsymbol{Q}_i - \lambda \cdot \boldsymbol{Q}_\ell$ 消去 x^s 次项 (为什么 $\mathrm{type}(\widetilde{\boldsymbol{Q}}_i) = i$?)
  - 对于 ℓ, 必须有 $\widetilde{\boldsymbol{Q}}_\ell = x \cdot \boldsymbol{Q}_\ell$. (总得牺牲一个)
- 正确性: 考虑消元.
- 时间复杂度: $\mathcal{O}(m^3 d^2) = \mathcal{O}(m\sigma^2)$. [Derksen 1994]

极小基的维护

- 按照 s 从小到大的顺序逐渐维护 V_s 的一组极小基.
- V_0 是平凡情况, 有 $\boldsymbol{Q}_{ij} = [i = j]$.
- 从 V_s 推到 V_{s+1} 的极小基 $\{\widetilde{\boldsymbol{Q}}_i\}$:
  - 如果 $x^{s+1} \mid \boldsymbol{Q}_i \cdot \boldsymbol{A}$, 可以直接保留, $\widetilde{\boldsymbol{Q}}_i = \boldsymbol{Q}_i$.
  - 否则存在 x^s 次项, 设 ℓ 是这种情况的 i 中按照 $(\deg, \mathrm{type})$ 的字典序比较下最小的那个.
  - 对于这样的 $i \neq \ell$, 通过 $\widetilde{\boldsymbol{Q}}_i = \boldsymbol{Q}_i - \lambda \cdot \boldsymbol{Q}_\ell$ 消去 x^s 次项 (为什么 $\mathrm{type}(\widetilde{\boldsymbol{Q}}_i) = i$?)
  - 对于 ℓ, 必须有 $\widetilde{\boldsymbol{Q}}_\ell = x \cdot \boldsymbol{Q}_\ell$. (总得牺牲一个)
- 正确性: 考虑消元.
- 时间复杂度: $\mathcal{O}(m^3 d^2) = \mathcal{O}(m\sigma^2)$. [Derksen 1994]
- 改进成一般情况: 把 deg 的定义改为 $\deg(\boldsymbol{P}) = \max_i \{\deg(\boldsymbol{P}_i) - d_i\}$.

极小基的维护

- 按照 s 从小到大的顺序逐渐维护 V_s 的一组极小基.
- V_0 是平凡情况, 有 $\boldsymbol{Q}_{ij} = [i = j]$.
- 从 V_s 推到 V_{s+1} 的极小基 $\{\widetilde{\boldsymbol{Q}}_i\}$:
  - 如果 $x^{s+1} \mid \boldsymbol{Q}_i \cdot \boldsymbol{A}$, 可以直接保留, $\widetilde{\boldsymbol{Q}}_i = \boldsymbol{Q}_i$.
  - 否则存在 x^s 次项, 设 ℓ 是这种情况的 i 中按照 $(\deg, \mathrm{type})$ 的字典序比较下最小的那个.
  - 对于这样的 $i \neq \ell$, 通过 $\widetilde{\boldsymbol{Q}}_i = \boldsymbol{Q}_i - \lambda \cdot \boldsymbol{Q}_\ell$ 消去 x^s 次项 (为什么 $\mathrm{type}(\widetilde{\boldsymbol{Q}}_i) = i$?)
  - 对于 ℓ, 必须有 $\widetilde{\boldsymbol{Q}}_\ell = x \cdot \boldsymbol{Q}_\ell$. (总得牺牲一个)
- 正确性: 考虑消元.
- 时间复杂度: $\mathcal{O}(m^3 d^2) = \mathcal{O}(m\sigma^2)$. [Derksen 1994]
- 改进成一般情况: 把 deg 的定义改为 $\deg(\boldsymbol{P}) = \max_i\{\deg(\boldsymbol{P}_i) - d_i\}$.
- 改进复杂度: 将 HalfGCD 的思想应用到上述过程!

大炮现状

定义 (Hermite 标准型)

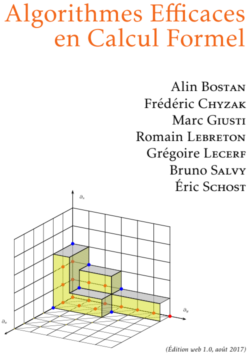
任何一个多项式矩阵 $\boldsymbol{A} \in \mathbb{F}[x]^{m \times m}$, 存在 $\boldsymbol{H} = \boldsymbol{A}\boldsymbol{U}$ $(\det(\boldsymbol{U}) \in \mathbb{F}^{\times})$ 使得

- $\boldsymbol{H}$ 是下三角矩阵.
- $\deg(\boldsymbol{H}_{ij}) < \deg(\boldsymbol{H}_{ii})$.

- 在 $\widetilde{\mathcal{O}}(m^{\omega} \deg(\boldsymbol{A}))$ 时间内计算 Hermite 标准型和 $\det(\boldsymbol{A})$. ($\deg(\boldsymbol{A})$ 是 “平均次数”!) [Labahn–Neiger–Zhou 2012]
- Popov 标准型, $\boldsymbol{s}$-约化型, ...
- Hermite–Padé 逼近, 但是从 $\bmod x^{\sigma}$ 改成 $\bmod M(x)$.
- 多项式矩阵的 “gcd”.
- ...

很多算法在 sage 中有实现.

延伸阅读



- von zur Gathen & Gerhard: 经典之作.
- AECF: 成书时间较新, 有网络版, 但是是法语.

定义

定义 (整式递推)

对于一个数列 $\{a_n\}_{n\geq 0}$, 我们称其为 **整式递推**(P-Recursive) 的, 当且仅当存在多项式 $p_0(x), p_1(x), \ldots, p_m(x)$ 使得, 对于 $n \geq m$, 有

$$p_0(n)a_n + p_1(n)a_{n-1} + \cdots + p_m(n)a_{n-m} = 0. \tag{68}$$

定义

定义 (整式递推)

对于一个数列 $\{a_n\}_{n\geq 0}$, 我们称其为 $\overset{\text{P-Recursive}}{\textbf{整式递推}}$ 的, 当且仅当存在多项式 $p_0(x), p_1(x), \ldots, p_m(x)$ 使得, 对于 $n \geq m$, 有

$$p_0(n)a_n + p_1(n)a_{n-1} + \cdots + p_m(n)a_{n-m} = 0. \tag{68}$$

定义 (微分有限)

对于一个生成函数 $A(x)$, 我们称其为 $\overset{\text{D-Finite}}{\textbf{微分有限}}$ 的, 当且仅当存在多项式 $f_0(x), \ldots, f_m(x)$, 有

$$f_0(x)A(x) + f_1(x)A'(x) + \cdots + f_m(x)A^{(m)}(x) = 0. \tag{69}$$

定义

定义 (整式递推)

对于一个数列 $\{a_n\}_{n\geq 0}$, 我们称其为 **整式递推**(P-Recursive) 的, 当且仅当存在多项式 $p_0(x), p_1(x), \ldots, p_m(x)$ 使得, 对于 $n \geq m$, 有

$$p_0(n)a_n + p_1(n)a_{n-1} + \cdots + p_m(n)a_{n-m} = 0. \tag{68}$$

定义 (微分有限)

对于一个生成函数 $A(x)$, 我们称其为 **微分有限**(D-Finite) 的, 当且仅当存在多项式 $f_0(x), \ldots, f_m(x)$, 有

$$f_0(x)A(x) + f_1(x)A'(x) + \cdots + f_m(x)A^{(m)}(x) = 0. \tag{69}$$

定理: $\{a_n\}$ 整式递推 $\iff A(x)$ 微分有限.

为什么要研究整式递推?

- 大量的组合计数问题的数列最终都被发现是整式递推的.

为什么要研究整式递推?

- 大量的组合计数问题的数列最终都被发现是整式递推的.
- 以 k-正则图计数为例.
- 之前的思考题里提到了, $k=2$ 的时候生成函数形如

$$A(x)=\exp\left(-\frac{x}{2}-\frac{x^2}{4}\right)\frac{1}{\sqrt{1-x}}, \tag{70}$$

- 微分有限的生成函数关于各种运算 **具有良好的封闭性**, 我们之后会看到有办法直接证明上述序列是整式递推的.

为什么要研究整式递推?

- 大量的组合计数问题的数列最终都被发现是整式递推的.
- 以 k-正则图计数为例.
- 之前的思考题里提到了, $k=2$ 的时候生成函数形如

$$A(x)=\exp\left(-\frac{x}{2}-\frac{x^2}{4}\right)\frac{1}{\sqrt{1-x}}, \tag{70}$$

- 微分有限的生成函数关于各种运算 **具有良好的封闭性**, 我们之后会看到有办法直接证明上述序列是整式递推的.
- 随后, $k=3$ 和 $k=4$ 的情况也被证明是整式递推的, 这一方法并非出于对于正则图组合结构的归纳. [Goulden–Jackson 1986]
- 之后, 任意 k 的情况也被证明. [Gessel 1990] 他们的证明方法源于发展了 **无穷元对称微分有限生成函数** 的理论.

线性代数 101

- 域 $\mathbb{F}$ 上的一个线性空间 V 是一个集合, 配备加法和对 $\mathbb{F}$ 的数乘运算, 满足线性性 ($\alpha,\beta\in\mathbb{F}$, $u,v\in V$):
  - $\alpha\cdot(u+v)=\alpha\cdot u+\alpha\cdot v$,
  - $(\alpha+\beta)\cdot u=\alpha\cdot u+\beta\cdot v$,
  - $\alpha\cdot(\beta\cdot v)=(\alpha\beta)\cdot v$.
- 我们会用到的线性空间: 多项式 $\mathbb{F}[x]$, 形式幂级数 $\mathbb{F}[\![x]\!]$, 形式 Laurent 级数 $\mathbb{F}(\!(x)\!)$, 数列 $\mathbb{F}^{\mathbb{Z}_{\geq 0}}$, ...
- 线性变换 $T: V\to W$ 是一个函数, 保持线性性:
  - $T(u+\alpha v)=Tu+\alpha Tv$.
- 我们关心的运算 ∂ 在 $\mathbb{F}[x]$, $\mathbb{F}[\![x]\!]$ 和 $\mathbb{F}(\!(x)\!)$ 都是 $\mathbb{F}$-线性的.
- 每个线性空间都有个维数, 是它的基的大小, 如果基的大小是有限的, 称是有限维线性空间 ($\dim V=n<\infty$).
- 比如 $\mathbb{F}[x]$ 不是有限维的, $\mathbb{F}^n$ 是 n 维的, 次数 $\leq n$ 的多项式是 $n+1$ 维的.

微分有限的线性空间表述

- 有理分式 $\mathbb{F}(x)$ 由于可以做除法, 所以它是个域.
- $\mathbb{F}(\!(x)\!)$ 不仅是 $\mathbb{F}$-线性空间, 还具有 $\mathbb{F}(x)$-线性结构!
  - 商 $\mathbb{F}[x]^{-1}\mathbb{F}[\![x]\!] \hookrightarrow \mathbb{F}(\!(x)\!)$ 容易识别, 这一嵌入进一步是同构, 但在多元情况并不典范.

定义 (微分有限)

一个生成函数 $F(x)$ 是 **微分有限** 的当且仅当

$$\mathscr{D}F := \operatorname{span}_{\mathbb{F}(x)}\left\{\partial^k F : k \in \mathbb{Z}_{\geq 0}\right\} \tag{71}$$

是有限维 $\mathbb{F}(x)$-线性空间.

- 这和我们之前的定义等价, 但更容易推广到多元情况.

微分有限的基本性质

定理

若 $F(x), G(x)$ 微分有限, 则 $F+G$ 微分有限.

证明.

令 $\{\alpha_1, \ldots, \alpha_n\}$ 为 $\mathscr{D}F$ 的基, $\{\beta_1, \ldots, \beta_m\}$ 为 $\mathscr{D}G$ 的基, 那么

$$\partial^k (F+G) = \partial^k F + \partial^k G = \sum a_i \alpha_i + \sum b_i \beta_i. \tag{72}$$

故 $\dim_{\mathbb{F}(x)} \mathscr{D}(F+G) \le n+m < \infty$. □

微分有限的基本性质

定理

若 $F(x), G(x)$ 微分有限, 则 FG 微分有限.

证明.

令 $\{\alpha_1, \ldots, \alpha_n\}$ 为 $\mathscr{D}F$ 的基, $\{\beta_1, \ldots, \beta_m\}$ 为 $\mathscr{D}G$ 的基, 那么 $\partial^k(FG)$ 被 $\partial^i F \partial^j G$ 线性表出, 因此被 $\{\alpha_i \beta_j\}$ 线性表出.
故 $\dim_{\mathbb{F}(x)} \mathscr{D}(FG) \le nm < \infty$. □

微分有限的基本性质

定理

若 $F(x), G(x)$ 微分有限, 则 $F+G$ 微分有限.

定理

若 $F(x), G(x)$ 微分有限, 则 FG 微分有限.

- 如何在计算上得到之前定义的对应的方程?

微分有限的基本性质

定理

若 $F(x), G(x)$ 微分有限, 则 $F+G$ 微分有限.

定理

若 $F(x), G(x)$ 微分有限, 则 FG 微分有限.

- 如何在计算上得到之前定义的对应的方程?
- F/G 并不一定是微分有限的:

$$\frac{x}{\exp x - 1}. \tag{73}$$

作为一个关心计算的人, 我为什么要研究整式递推?

- ∂ 连同四则运算, 很大程度上勾勒了我们有哪些工具计算一个序列.
- 生成函数的层级:

$$\text{有理} \subset \text{代数} \subset \text{微分有限} \subset \overset{\text{D-Algebraic}}{\text{微分代数}} \subset \mathbb{F}[\![x]\!]$$

作为一个关心计算的人, 我为什么要研究整式递推?

- ∂ 连同四则运算, 很大程度上勾勒了我们有哪些工具计算一个序列.
- 生成函数的层级:

$$\text{有理} \subset \text{代数} \subset \text{微分有限} \subset \overset{\text{D-Algebraic}}{\text{微分代数}} \subset \mathbb{F}[\![x]\!]$$

  - 如果是有理函数, 我们可以 $O(\log N)$ 就计算出第 N 项, 相当快速.
  - 如果微分有限, 我们可以 $O(\mathsf{M}(\sqrt{N}))$ 就计算出第 N 项 [Chudnovsky–Chudnovsky 1988] [Bostan–Gaudry–Schost 2007], 或者 $O(N)$ 计算出前 N 项.
  - 如果微分代数, 我们至少可以用半在线卷积的方法 $O(\mathsf{R}(N))$ 计算出前 N 项.

题外话: 疑难数列之整数拆分

- 整数拆分的生成函数

$$F(x) = \prod_{n=1}^{\infty} \frac{1}{1-x^n}, \tag{74}$$

题外话: 疑难数列之整数拆分

- 整数拆分的生成函数

$$F(x)=\prod_{n=1}^{\infty}\frac{1}{1-x^n}, \tag{74}$$

看起来应该是非常困难的序列, 但它居然 **是** 微分代数的!

$$4F^3F''+5xF^3F'''+x^2F^3F^{(4)}-16F^2F'^2-15xF^2F'F''+20x^2F^2F'F'''$$
$$-39x^2F^2F''^2+10xFF'^3+12x^2FF'^2F''+6x^2F'^4=0. \tag{75}$$

题外话: 疑难数列之整数拆分

- 整数拆分的生成函数

$$F(x) = \prod_{n=1}^{\infty} \frac{1}{1-x^n}, \tag{74}$$

看起来应该是非常困难的序列, 但它居然 **是** 微分代数的!

$$4F^3F'' + 5xF^3F''' + x^2F^3F^{(4)} - 16F^2F'^2 - 15xF^2F'F'' + 20x^2F^2F'F''' \\ - 39x^2F^2F''^2 + 10xFF'^3 + 12x^2FF'^2F'' + 6x^2F'^4 = 0. \tag{75}$$

- 深层原因或许要涉及 模形式 (Modular Form), 这是现代数学的重要分支之一, 远远超出了本次讲故事的狩猎范围.

题外话: 疑难数列之禁子结构排列

- 给定一个有限个排列构成的集合 $\mathscr{F}$, 记 $\mathrm{A}_n(\mathscr{F})$ 是所有没有出现过 $\mathscr{F}$ 中结构的 n 阶排列的数量.

题外话: 疑难数列之禁子结构排列

- 给定一个有限个排列构成的集合 $\mathscr{F}$, 记 $\mathrm{A}_n(\mathscr{F})$ 是所有没有出现过 $\mathscr{F}$ 中结构的 n 阶排列的数量.
- 例如 $\mathrm{A}_n(\{123\})$ 就是 Catalan 数 C_n.

题外话: 疑难数列之禁子结构排列

- 给定一个有限个排列构成的集合 $\mathscr{F}$, 记 $\mathsf{A}_n(\mathscr{F})$ 是所有没有出现过 $\mathscr{F}$ 中结构的 n 阶排列的数量.
- 例如 $\mathsf{A}_n(\{123\})$ 就是 Catalan 数 C_n.
- 但到了形态有 4 阶排列, 就已经出现了没法解决的问题了, 例如 $\mathsf{A}_n(\{1324\})$.

题外话: 疑难数列之禁子结构排列

- 给定一个有限个排列构成的集合 $\mathscr{F}$, 记 $\mathsf{A}_n(\mathscr{F})$ 是所有没有出现过 $\mathscr{F}$ 中结构的 n 阶排列的数量.
- 例如 $\mathsf{A}_n(\{123\})$ 就是 Catalan 数 C_n.
- 但到了形态有 4 阶排列, 就已经出现了没法解决的问题了, 例如 $\mathsf{A}_n(\{1324\})$.
  - 人们现在 **不知道** 如何在多项式时间内计算 $\mathsf{A}_n(\{1324\})$. 尽管人们现在已经确认计算 $\mathsf{A}_n(\mathscr{F})$ **很可能**是困难的. (如果存在多项式时间算法计算 $\mathsf{A}_n(\mathscr{F}) \bmod 2$, 那么 EXP = ⊕EXP. [Garrabrant–Pak 2015])

题外话: 疑难数列之禁子结构排列

- 给定一个有限个排列构成的集合 $\mathscr{F}$, 记 $\mathrm{A}_n(\mathscr{F})$ 是所有没有出现过 $\mathscr{F}$ 中结构的 n 阶排列的数量.
- 例如 $\mathrm{A}_n(\{123\})$ 就是 Catalan 数 C_n.
- 但到了形态有 4 阶排列, 就已经出现了没法解决的问题了, 例如 $\mathrm{A}_n(\{1324\})$.
    - 人们现在 **不知道** 如何在多项式时间内计算 $\mathrm{A}_n(\{1324\})$. 尽管人们现在已经确认计算 $\mathrm{A}_n(\mathscr{F})$ **很可能**是困难的. (如果存在多项式时间算法计算 $\mathrm{A}_n(\mathscr{F}) \bmod 2$, 那么 $\mathsf{EXP} = \oplus\mathsf{EXP}$. [Garrabrant–Pak 2015])
    - 人们现在 **不知道** 如何证明 $\mathrm{A}_n(\{1324\})$ 并非整式递推. 尽管已经知道存在 $\mathscr{F} \subset \mathfrak{S}_{80}$ 使得 $\mathrm{A}_n(\mathscr{F})$ 不是整式递推. [Garrabrant–Pak 2015]

题外话: 疑难数列之禁子结构排列

- 给定一个有限个排列构成的集合 $\mathscr{F}$, 记 $\mathsf{A}_n(\mathscr{F})$ 是所有没有出现过 $\mathscr{F}$ 中结构的 n 阶排列的数量.
- 例如 $\mathsf{A}_n(\{123\})$ 就是 Catalan 数 C_n.
- 但到了形态有 4 阶排列, 就已经出现了没法解决的问题了, 例如 $\mathsf{A}_n(\{1324\})$.
  - 人们现在 **不知道** 如何在多项式时间内计算 $\mathsf{A}_n(\{1324\})$. 尽管人们现在已经确认计算 $\mathsf{A}_n(\mathscr{F})$ **很可能**是困难的. (如果存在多项式时间算法计算 $\mathsf{A}_n(\mathscr{F}) \bmod 2$, 那么 $\mathsf{EXP} = \oplus\mathsf{EXP}$. [Garrabrant–Pak 2015])
  - 人们现在 **不知道** 如何证明 $\mathsf{A}_n(\{1324\})$ 并非整式递推. 尽管已经知道存在 $\mathscr{F} \subset \mathfrak{S}_{80}$ 使得 $\mathsf{A}_n(\mathscr{F})$ 不是整式递推. [Garrabrant–Pak 2015]
  - 人们现在甚至不能证明 $\mathsf{A}_n(\{1324\})$ 的渐进行为, 只能根据 $n \le 50$ 的数值结果做出猜测: [Conway–Guttmann–Zinn-Justin 2017]

$$\mathsf{A}_n(\{1324\}) \sim C \cdot \lambda^n \mu^{\sqrt{n}} n^{\alpha}. \tag{76}$$

代数幂级数

定义 (代数幂级数)

$F(x) \in \mathbb{F}[\![x]\!]$ 是 **代数** 的当且仅当有多项式方程 $P \in \mathbb{F}[X,T]$ 满足

$$P(x,F) = 0. \tag{77}$$

代数幂级数关于除法是封闭的!

定理

代数幂级数构成域, 也即 $F+G, FG, F/G$ 都是代数幂级数.

加法和乘法略去, 仅勾勒除法的封闭性.

证明.

如果 $P(x,F) = 0$, 写作 $P(X,T) = \prod_\alpha (T - \alpha(x))$, 那么

$$\prod_\alpha (T - 1/\alpha) = 0 \implies \prod_\alpha (\alpha T - 1) = 0 \implies T^{\deg P} P(x, 1/T) = 0. \tag{78}$$

$\square$

代数幂级数的微分有限性

定理

如果 $u(x)$ 是代数的, 那么它是微分有限的.

证明 (勾勒).

设 u 满足一个次数为 n 的代数方程 $P(x,u)=0$, 求导可以得到 $\mathscr{D}u \subset \mathrm{span}_{\mathbb{F}(x)}\{u^k : 0 \le k < n\}$. □

后者其实就是代数扩张 $\mathbb{K} := \mathbb{F}(x)(u)$.

定理

如果 $F(x)$ 是微分有限的, $u(x)$ 是代数的, 那么 $F(u(x))$ 是微分有限的.

证明 (勾勒).

只需证明 $\partial^k(F \circ u)$ 张成的是有限维 $\mathbb{K}$-空间. □

大炮开火!

- 2-正则图构成的组合类, 其指数型生成函数是

$$A(x) = \exp\left(-\frac{x}{2} - \frac{x^2}{4}\right)\frac{1}{\sqrt{1-x}}. \tag{79}$$

大炮开火!

- 2-正则图构成的组合类, 其指数型生成函数是

$$A(x) = \exp\left(-\frac{x}{2} - \frac{x^2}{4}\right)\frac{1}{\sqrt{1-x}}. \tag{79}$$

- $\exp\left(-\frac{x}{2} - \frac{x^2}{4}\right)$ 是微分有限的, 因为

大炮开火!

- 2-正则图构成的组合类, 其指数型生成函数是

$$A(x) = \exp\left(-\frac{x}{2} - \frac{x^2}{4}\right)\frac{1}{\sqrt{1-x}}. \tag{79}$$

- $\exp\left(-\frac{x}{2} - \frac{x^2}{4}\right)$ 是微分有限的, 因为
  - $\exp x$ 是微分有限的.

大炮开火!

- 2-正则图构成的组合类, 其指数型生成函数是

$$A(x) = \exp\left(-\frac{x}{2} - \frac{x^2}{4}\right) \frac{1}{\sqrt{1-x}}. \tag{79}$$

- $\exp\left(-\frac{x}{2} - \frac{x^2}{4}\right)$ 是微分有限的, 因为
  - $\exp x$ 是微分有限的.
  - $-\frac{x}{2} - \frac{x^2}{4}$ 是有理的, 所以是代数的.

大炮开火!

- 2-正则图构成的组合类, 其指数型生成函数是

$$A(x) = \exp\left(-\frac{x}{2} - \frac{x^2}{4}\right)\frac{1}{\sqrt{1-x}}. \tag{79}$$

- $\exp\left(-\frac{x}{2} - \frac{x^2}{4}\right)$ 是微分有限的, 因为
  - $\exp x$ 是微分有限的.
  - $-\frac{x}{2} - \frac{x^2}{4}$ 是有理的, 所以是代数的.
- $\frac{1}{\sqrt{1-x}}$ 是微分有限的, 因为它是代数的.

大炮开火!

- 2-正则图构成的组合类, 其指数型生成函数是

$$A(x) = \exp\left(-\frac{x}{2} - \frac{x^2}{4}\right) \frac{1}{\sqrt{1-x}}. \tag{79}$$

- $\exp\left(-\frac{x}{2} - \frac{x^2}{4}\right)$ 是微分有限的, 因为
  - $\exp x$ 是微分有限的.
  - $-\frac{x}{2} - \frac{x^2}{4}$ 是有理的, 所以是代数的.
- $\frac{1}{\sqrt{1-x}}$ 是微分有限的, 因为它是代数的.
- 因此 $A(x)$ 是微分有限的.

更多代数幂级数

二元分式

$$\frac{1}{1-x-y} = \sum_{n,m\geq 0} \binom{n+m}{n} x^n y^m \tag{80}$$

的对角线是

$$\sum_{n=0}^{\infty} \binom{2n}{n} x^n = \frac{1}{\sqrt{1-4x}}, \tag{81}$$

根据后见之明, 可以用广义二项式, 但如何直接得到?

有理分式的对角线

定理

对于有理分式 $Q(x,y)$, 其对角线 $\sum_{n=0}^{\infty}([x^n y^n]Q)T^n$ 是代数的.

证明 (勾勒).

欲提取 $Q(S,T/S)$ 的 S^0 次项, 分式分解展开成

$$Q(S,T/S) = \sum_{\alpha_i,\beta_i,n_i} \frac{\alpha_i}{(S-\beta_i)^{n_i}}, \tag{82}$$

提取其常数项, 一些代数函数的和, 还是代数的. □

为什么有的向 $+\infty$ 方向展开, 有的向 $-\infty$ 方向展开?

代数方程

- 在 $\mathbb{Z}_{\geq 0}$ 上游走, 有 a, b, c, d 种方法走 $+1, +2, -1, -2$ 的位移, 有多少种走法走 n 步从原点回到原点?

代数方程

- 在 $\mathbb{Z}_{\geq 0}$ 上游走, 有 a,b,c,d 种方法走 $+1,+2,-1,-2$ 的位移, 有多少种走法走 n 步从原点回到原点?
- 设 $F_{0,0},F_{1,0},F_{0,1},F_{1,1}$, 分别表示最初和最终欠几步.

$$F_{0,0} = \frac{1}{1-x^2(acF_{0,0}+adF_{1,0}+bcF_{0,1}+cdF_{1,1})}, \tag{83}$$

$$F_{1,0} = x(cF_{0,0}+dF_{0,1})\cdot F_{0,0}, \tag{84}$$

$$F_{0,1} = x(aF_{0,0}+bF_{1,0})\cdot F_{0,0}, \tag{85}$$

$$F_{1,1} = F_{0,0}+x(cF_{0,0}+dF_{0,1})\cdot F_{0,0}\cdot x(aF_{0,0}+bF_{1,0}). \tag{86}$$

代数方程

- 在 $\mathbb{Z}_{\geq 0}$ 上游走, 有 a,b,c,d 种方法走 $+1,+2,-1,-2$ 的位移, 有多少种走法走 n 步从原点回到原点?
- 设 $F_{0,0},F_{1,0},F_{0,1},F_{1,1}$, 分别表示最初和最终欠几步.

$$F_{0,0} = \frac{1}{1-x^2(acF_{0,0}+adF_{1,0}+bcF_{0,1}+cdF_{1,1})}, \tag{83}$$

$$F_{1,0} = x(cF_{0,0}+dF_{0,1})\cdot F_{0,0}, \tag{84}$$

$$F_{0,1} = x(aF_{0,0}+bF_{1,0})\cdot F_{0,0}, \tag{85}$$

$$F_{1,1} = F_{0,0}+x(cF_{0,0}+dF_{0,1})\cdot F_{0,0}\cdot x(aF_{0,0}+bF_{1,0}). \tag{86}$$

- 数值实验表明 $F_{0,0}$ 满足一个次数 ≤ 6 的代数方程.

代数方程

- $F_{0,0}$ 满足一个次数 ≤ 6 的代数方程.
- 我们有四个未知量, 列了四个方程.
- 记 $\mathbb{K} = \mathbb{F}(x, a, b, c, d)$.

代数方程

- $F_{0,0}$ 满足一个次数 ≤ 6 的代数方程.
- 我们有四个未知量, 列了四个方程.
- 记 $\mathbb{K} = \mathbb{F}(x, a, b, c, d)$.

Bézout 定理

"一般" 的 n 个 n 元多项式方程组, 次数分别为 $d_1, \ldots, d_n$, 在 $\overline{\mathbb{K}}$ 上有 $d_1 \cdots d_n$ 个解.

代数方程

- $F_{0,0}$ 满足一个次数 ≤ 6 的代数方程.
- 我们有四个未知量, 列了四个方程.
- 记 $\mathbb{K} = \mathbb{F}(x,a,b,c,d)$.

Bézout 定理

"一般" 的 n 个 n 元多项式方程组, 次数分别为 $d_1,\ldots,d_n$, 在 $\overline{\mathbb{K}}$ 上有 $d_1 \cdots d_n$ 个解.

- 所以容易用 Hermite–Padé 逼近猜出最小多项式.

代数方程

- $F_{0,0}$ 满足一个次数 ≤ 6 的代数方程.
- 我们有四个未知量, 列了四个方程.
- 记 $\mathbb{K} = \mathbb{F}(x, a, b, c, d)$.

Bézout 定理

"一般" 的 n 个 n 元多项式方程组, 次数分别为 $d_1, \ldots, d_n$, 在 $\overline{\mathbb{K}}$ 上有 $d_1 \cdots d_n$ 个解.

- 所以容易用 Hermite–Padé 逼近猜出最小多项式.
- 一个构成证明的计算过程可以考虑使用 Gröbner 基, 给 $\mathbb{K}[F_{0,0}, F_{1,0}, F_{0,1}, F_{1,1}]$ 设置主元的 "顺序" $F_{0,0} \prec F_{1,0} \prec F_{0,1} \prec F_{1,1}$, 最后会消出一个只含 $F_{0,0}$ 的多项式, 这就给出了 $F_{0,0}$ 满足的一个多项式方程.

作为一个关心封闭形式的人, 我为什么要研究整式递推?

- 首先, 大部分数列如果有比较 **正常** 的封闭形式, 这个封闭形式也很多时候是整式递推的.
- 如何证明 $A=B$? 其中 A,B 都是组合求和式.
  1. 在计算机的辅助下, 设计对应的消元算法, 分别得到 A,B 的一个整式递推式.
  2. 找到它们公共满足的整式递推式.
  3. 暴力验证前面充分多项, 说明两个数列相等.

Gessel 游走

- 定义 $q(i,j;n)$ 是满足如下要求的 n 步游走, 从 $(0,0)$ 到达 (i,j) 的方案数:
  - 途中位置只能在 $\mathbb{Z}^2_{\geq 0}$ 上.
  - 每一步只能是 $\{\leftarrow, \rightarrow, \nearrow, \swarrow\}$.

Gessel 游走

- 定义 $q(i,j;n)$ 是满足如下要求的 n 步游走, 从 $(0,0)$ 到达 (i,j) 的方案数:
  - 途中位置只能在 $\mathbb{Z}^2_{\geq 0}$ 上.
  - 每一步只能是 $\{\leftarrow, \rightarrow, \nearrow, \swarrow\}$.
- Gessel 猜想 $q(0,0;2n) = 16^n \frac{(5/6)_n (1/2)_n}{(2)_n (5/3)_n}$, 在计算机的辅助下被证明. [Kauers–Koutschan–Zeilberger 2008]

Gessel 游走

- 定义 $q(i,j;n)$ 是满足如下要求的 n 步游走, 从 $(0,0)$ 到达 (i,j) 的方案数:
  - 途中位置只能在 $\mathbb{Z}^2_{\geq 0}$ 上.
  - 每一步只能是 $\{\leftarrow, \rightarrow, \nearrow, \swarrow\}$.
- Gessel 猜想 $q(0,0;2n) = 16^n \frac{(5/6)_n (1/2)_n}{(2)_n (5/3)_n}$, 在计算机的辅助下被证明. [Kauers–Koutschan–Zeilberger 2008]
- 过了很久才得到一个完全由人类完成的证明, 并且过程并不初等, 用到了椭圆函数. [Bostan–Kurkova–Raschel 2017]

Gessel 游走

- 定义 $q(i,j;n)$ 是满足如下要求的 n 步游走, 从 $(0,0)$ 到达 (i,j) 的方案数:
  - 途中位置只能在 $\mathbb{Z}^2_{\geq 0}$ 上.
  - 每一步只能是 $\{\leftarrow, \rightarrow, \nearrow, \swarrow\}$.
- Gessel 猜想 $q(0,0;2n) = 16^n \frac{(5/6)_n (1/2)_n}{(2)_n (5/3)_n}$, 在计算机的辅助下被证明. [Kauers–Koutschan–Zeilberger 2008]
- 过了很久才得到一个完全由人类完成的证明, 并且过程并不初等, 用到了椭圆函数. [Bostan–Kurkova–Raschel 2017]

Proof of the algebraicity of the trivariate GF. We start by proving the algebraicity of $Q(0, y)$ as a function of y, z. We consider the representation of $r_y(\omega)$ given in Theorem 3 and apply eight times the addition theorem (P4) for ζ-functions, namely (for suitable values of $k \in \mathbf{Z}$ that can be deduced from (21))

$$\zeta_{1,3}(\omega - k\omega_2/8) = \zeta_{1,3}(\omega) - \zeta_{1,3}(k\omega_2/8) + \frac{1}{2}\frac{\wp'_{1,3}(\omega) + \wp'_{1,3}(k\omega_2/8)}{\wp_{1,3}(\omega) - \wp_{1,3}(k\omega_2/8)}.$$

We then make the weighted sum of the eight identities above (corresponding to the good values of k in (21)); this way, we obtain

$$r_y(\omega) = U_1(\omega) + U_2 + U_3(\omega),$$

图: 证明一瞥

Gessel 游走

- 定义 $q(i,j;n)$ 是满足如下要求的 n 步游走, 从 $(0,0)$ 到达 (i,j) 的方案数:
  - 途中位置只能在 $\mathbb{Z}^2_{\geq 0}$ 上.
  - 每一步只能是 $\{\leftarrow, \rightarrow, \nearrow, \swarrow\}$.
- Gessel 猜想 $q(0,0;2n) = 16^n \frac{(5/6)_n (1/2)_n}{(2)_n (5/3)_n}$, 在计算机的辅助下被证明. [Kauers–Koutschan–Zeilberger 2008]
- 过了很久才得到一个完全由人类完成的证明, 并且过程并不初等, 用到了椭圆函数. [Bostan–Kurkova–Raschel 2017]
- 大结局: 在计算机的辅助下被证明, 整个 3 维序列的生成函数 $G(x,y;t)$ 是 **代数** 的. [Bostan–Kauers 2010]
  - 需要计算前 $n \leq 1200$ 的所有项, 大概 1.5×10^9 项.
  - 求出的最小多项式 $P(G(x,y;t);x,y,t) = 0$ 的系数有 10^{11} 项, 需要 30 Gb 才能存下!

多元整式递推: 如何定义?

- Zeilberger 最初的尝试: 一个 $f:\mathbb{Z}_{\geq 0}^{n}\to\mathbb{F}$ 的 n 元数列, 对于每个 $1\leq i\leq n$, 存在一组多项式 $P_j^{[i]}$ $(1\leq j\leq r_i)$, 使得满足递推式

$$\sum_{j=0}^{r_i} P_j^{[i]}(\boldsymbol{m}) f(m_1,\ldots,m_i-j,\ldots,m_n)=0. \tag{87}$$

多元整式递推: 如何定义?

- Zeilberger 最初的尝试: 一个 $f:\mathbb{Z}_{\geq 0}^n \to \mathbb{F}$ 的 n 元数列, 对于每个 $1\leq i\leq n$, 存在一组多项式 $P_j^{[i]}$ $(1\leq j\leq r_i)$, 使得满足递推式

$$\sum_{j=0}^{r_i} P_j^{[i]}(\boldsymbol{m}) f(m_1,\ldots,m_i-j,\ldots,m_n)=0. \tag{87}$$

- 这个定义有严重的问题! Stanley 给出了一个例子: 对于 $f(n,m)(n^2-m)=0$ 这个方程, 有一组解

$$\sum_{n=0}^{\infty} x^n y^{n^2}, \tag{88}$$

但是这个函数的性质相当复杂, 是我们想要排除的. [Gessel 90]

多元微分有限

定义 (多元微分有限)

一个 n 元生成函数 $F(x_1,\ldots,x_n) \in \mathbb{F}[\![x_1,\ldots,x_n]\!]$ 是 **微分有限** 的当且仅当

$$\mathscr{D}F := \operatorname{span}_{\mathbb{F}(x_1,\ldots,x_n)} \left\{ \partial_{x_1}^{k_1} \cdots \partial_{x_n}^{k_n} F : \boldsymbol{k} \in \mathbb{Z}_{\geq 0}^n \right\} \tag{89}$$

是有限维 $\mathbb{F}(x_1,\ldots,x_n)$-线性空间.

- 等等, 如何赋予 $\mathbb{F}[\![x_1,\ldots,x_n]\!]$ 以 $\mathbb{F}(x_1,\ldots,x_n)$-线性结构? (考虑如何展开 $1/(x-y)$)
- 当然嵌入到 $\mathbb{F}(\!(x_1)\!)\cdots(\!(x_n)\!)$ 是一种办法, 不过这钦定了一个顺序, 并不典范.
- 一个重要的 **等价定义**: 放宽成对于每个 $1 \leq i \leq n$,

$$\dim \operatorname{span}_{\mathbb{F}(x_1,\ldots,x_n)} \left\{ \partial_{x_i}^k F : k \in \mathbb{Z}_{\geq 0} \right\} < \infty \tag{90}$$

多元微分有限的基本性质

定理

如果 $F, G \in \mathbb{F}[\![x_1, \ldots, x_n]\!]$ 是微分有限的, $u_1, \ldots, u_n \in \mathbb{F}[\![t_1, \ldots, t_m]\!]$ 是代数的, 那么以下生成函数微分有限:

- $F + G$.
- $F \cdot G$.
- 良定义的 $F(u_1, \ldots, u_n)$.

多元微分有限的基本性质

定理

如果 $F, G \in \mathbb{F}[\![x_1, \ldots, x_n]\!]$ 是微分有限的, $u_1, \ldots, u_n \in \mathbb{F}[\![t_1, \ldots, t_m]\!]$ 是代数的, 那么以下生成函数微分有限:

- $F + G$.
- $F \cdot G$.
- 良定义的 $F(u_1, \ldots, u_n)$.
- **对角线** ***[Lipshitz 1988]***

$$\sum_{i_1\ldots, i_{n-1} \in \mathbb{Z}_{\geq 0}} f_{i_1 \ldots i_{n-1} i_{n-1}} x_1^{i_1} \cdots x_{n-1}^{i_{n-1}}. \tag{91}$$

终结比赛的对角线 [Lipshitz 1988]

- 几乎所有正常的和式都是微分有限的!

$$c_{i,j} = \sum_k a_{i,k} b_{k,j}. \tag{92}$$

终结比赛的对角线 [Lipshitz 1988]

- 几乎所有正常的和式都是微分有限的!

$$c_{i,j} = \sum_k a_{i,k} b_{k,j}. \tag{92}$$

- 首先 $A(X,Y)B(Y',Z)$, 然后缩并 Y 和 Y', 然后带入 $Y = 1$.

证明勾勒: 对角线的微分有限性 [Lipshitz 1988]

为了使得呈现更加清晰, 我们只证明二元情况 ($F(x,y)$), 多元情况可以照猫画虎.

1. 换元为 $G = s^{-1}F(x/s,s)$, 这是关于 x,s "微分有限" 的, 不是形式幂级数, 但是仍然满足前述的 $\dim \mathscr{D} G < \infty$.
2. 证明存在非零解满足
$$\sum_{k,\ell} p_{k,\ell}(x)\partial_x^k \partial_s^\ell G = 0. \tag{93}$$
3. 设 o 是 ∂_s^ℓ 的系数不为零的最小的 ℓ, 那么上式的 s^{-o-1} 次项系数给出等式
$$\sum_k p_{k,o}(x)\partial_x^k([s^{-1}]G) = 0. \tag{94}$$
4. 说明 $[s^{-1}]G$ 也即 F 的对角线微分有限! $\square$

关键引理 [Lipshitz 1988]

欲证 $\sum_{k,\ell} p_{k,\ell}(x)\partial_x^k\partial_s^\ell G = 0$.

- 根据微分有限性条件, 得到多项式方程 ($\deg L = \ell$)

$$L(x,s)\partial_x^d G = O(s^d, \partial_x^{d-1})G, \tag{95}$$
$$L(x,s)\partial_s^d G = O(s^d, \partial_s^{d-1})G. \tag{96}$$

- 考虑一个大 N, 以及所有 $\alpha+\beta\le N$, 考虑 $L^N\partial_x^\alpha\partial_s^\beta G$, 通过上述方程不断约化为
$$L^N\partial_x^\alpha\partial_s^\beta G = O(s^{(d+\ell)N}, \partial_x^{d-1}, \partial_s^{d-1})G. \tag{97}$$
- 全体 α,β 一共有 $\Omega(N^2)$ 种选择, 但右侧的 $s^i\partial_x^j\partial_s^k$ 只有 $\mathcal{O}(N)$ 种情况, **所以当 N 充分大**, 一定可以将左侧 $\mathbb{F}(x)$-线性组合得到右侧为 0.
- 计算这种解的任务一般被称作计算 $\overset{\text{syzygy}}{\text{合冲}}$.

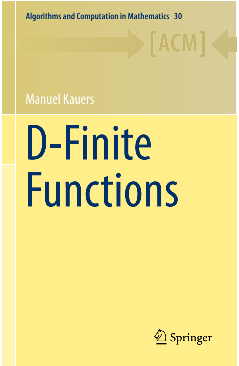
来不及讲的话题

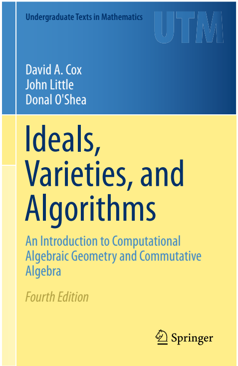
- 小模数 p?
  - 固定模数 p, 代数幂级数的单项求值都有 “数位 DP” 算法.
  - p-自动机和代数幂级数的等价性.
  - 整式递推除以 0?
- Weyl 代数 $\mathbb{F}[x,\partial]$ 和 Ore 代数 $\mathbb{F}[x_1,\ldots,x_n,\partial_1,\ldots,\partial_n]$?
  - 快速计算乘法 (矩阵乘法)?
  - 不交换的代数结构, 但是可以定义一个方向的 Euclid 算法和 gcd.
- q-整式递推?
  - 咬文嚼字: [二项式 / 整式递推 / 超几何级数 / ...] 的 $\overset{q\text{-analog}}{q\text{-类比}}$, 或者 q-[二项式 / 整式递推 / 超几何级数 / ...], 而不是单独说 “q-类比”?

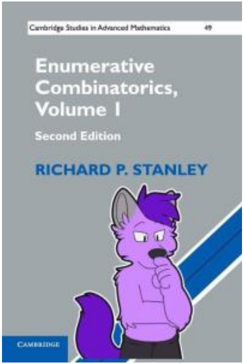
“还有许多问题我愿意告诉你们, 但是你们现在尚不能接受.”

— A. Кострикин, 代数学引论

延伸阅读


Algorithms and Computation in Mathematics 30
[ACM]
Manuel Kauers
D-Finite Functions
Springer


Undergraduate Texts in Mathematics
UTM
David A. Cox
John Little
Donal O'Shea
Ideals, Varieties, and Algorithms
An Introduction to Computational Algebraic Geometry and Commutative Algebra
Fourth Edition
Springer


Cambridge Studies in Advanced Mathematics 49
Enumerative Combinatorics, Volume I
Second Edition
RICHARD P. STANLEY
CAMBRIDGE

后日谈: 整式递推在 OI 中的未来

- “很多序列都是整式递推的”, 这是一个对于我们理解问题的正面消息, 同时也是对出题人的新考验.

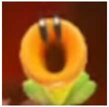
后日谈: 整式递推在 OI 中的未来

- “很多序列都是整式递推的”, 这是一个对于我们理解问题的正面消息, 同时也是对出题人的新考验.
- 出题人: 我动用了很多智慧, 最后得到了这个问题答案的递推式!

后日谈: 整式递推在 OI 中的未来

- “很多序列都是整式递推的”, 这是一个对于我们理解问题的正面消息, 同时也是对出题人的新考验.
- 出题人: 我动用了很多智慧, 最后得到了这个问题答案的递推式!
- 选手: 跑几项暴力, Min25 BM 直接秒了, 真简单!

后日谈: 整式递推在 OI 中的未来

- “很多序列都是整式递推的”, 这是一个对于我们理解问题的正面消息, 同时也是对出题人的新考验.
- 出题人: 我动用了很多智慧, 最后得到了这个问题答案的递推式!
- 选手: 跑几项暴力, Min25 BM 直接秒了, 真简单!
- 出题人: 

没有绝对的 “最小递推式”, 只有 Pareto 最优!

Example 1.36 The formal power series

$$\frac{1+x^5}{\sqrt{x+1}} + \frac{2x+3}{\sqrt{1-x}} + (3x^4 - 4x^3 + 8)\exp\left(\frac{x}{1-x}\right) = 12 + 11x + \tfrac{29}{2}x^2 + \cdots$$

satisfies a differential equation of order r with polynomial coefficients of degree d for every point (r, d) in the gray region in the figure below.

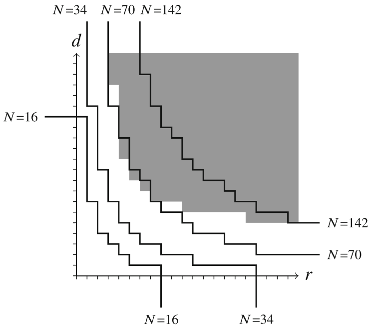


图: 递推式的长度–多项式次数的权衡 [Kauers 2023]

没有一劳永逸的方法

- 微分有限这一定义本身并不能完整捕捉 $\overset{\text{effective}}{\text{有效}}$ 的递推式.

没有一劳永逸的方法

- 微分有限这一定义本身并不能完整捕捉 $\overset{\text{effective}}{\text{有效}}$ 的递推式.
- 回归一个古老的启蒙问题: 给定多项式 $f(x)$, 求出 $f(x)^n$ 的各项系数.

没有一劳永逸的方法

- 微分有限这一定义本身并不能完整捕捉 $\overset{\text{effective}}{\text{有效}}$ 的递推式.
- 回归一个古老的启蒙问题: 给定多项式 $f(x)$, 求出 $f(x)^n$ 的各项系数.
  - 记 $g(x) = f(x)^n$, 那么可以利用 $g'f = nf'g$ 来递推.

没有一劳永逸的方法

- 微分有限这一定义本身并不能完整捕捉 $\overset{\text{effective}}{\text{有效}}$ 的递推式.
- 回归一个古老的启蒙问题: 给定多项式 $f(x)$, 求出 $f(x)^n$ 的各项系数.
  - 记 $g(x) = f(x)^n$, 那么可以利用 $g'f = nf'g$ 来递推.
  - 如果 $f(x)$ 不是低次多项式, 而是 **稀疏多项式**, 方法仍然奏效, 但难以用 Min25 BM 解决.

没有一劳永逸的方法

- 微分有限这一定义本身并不能完整捕捉 $\overset{\text{effective}}{\text{有效}}$ 的递推式.
- 回归一个古老的启蒙问题: 给定多项式 $f(x)$, 求出 $f(x)^n$ 的各项系数.
    - 记 $g(x) = f(x)^n$, 那么可以利用 $g'f = nf'g$ 来递推.
    - 如果 $f(x)$ 不是低次多项式, 而是 **稀疏多项式**, 方法仍然奏效, 但难以用 Min25 BM 解决.
    - **稀疏整式递推**, 需要理解操作原理才能得到 **有效** 递推式的问题.

没有一劳永逸的方法

- 微分有限这一定义本身并不能完整捕捉 $\overset{\text{effective}}{\text{有效}}$ 的递推式.
- 回归一个古老的启蒙问题: 给定多项式 $f(x)$, 求出 $f(x)^n$ 的各项系数.
  - 记 $g(x) = f(x)^n$, 那么可以利用 $g'f = nf'g$ 来递推.
  - 如果 $f(x)$ 不是低次多项式, 而是 **稀疏多项式**, 方法仍然奏效, 但难以用 Min25 BM 解决.
  - **稀疏整式递推**, 需要理解操作原理才能得到 **有效** 递推式的问题.
- **隐式整式递推**: 答案序列 $f(n)$ 本身并非整式递推, 但是计算某一项的时候具有整式递推的内核.

没有一劳永逸的方法

- 微分有限这一定义本身并不能完整捕捉 $\overset{\text{effective}}{\text{有效}}$ 的递推式.
- 回归一个古老的启蒙问题: 给定多项式 $f(x)$, 求出 $f(x)^n$ 的各项系数.
  - 记 $g(x) = f(x)^n$, 那么可以利用 $g'f = nf'g$ 来递推.
  - 如果 $f(x)$ 不是低次多项式, 而是 **稀疏多项式**, 方法仍然奏效, 但难以用 Min25 BM 解决.
  - **稀疏整式递推**, 需要理解操作原理才能得到 **有效** 递推式的问题.
- **隐式整式递推**: 答案序列 $f(n)$ 本身并非整式递推, 但是计算某一项的时候具有整式递推的内核.
  - 截取–Taylor–截取: $\mathscr{O}(n)$ 计算 Bernoulli 数 $B_n = [x^n/n!]\frac{x}{\exp x - 1}$.

没有一劳永逸的方法

- 微分有限这一定义本身并不能完整捕捉 $\overset{\text{effective}}{\text{有效}}$ 的递推式.
- 回归一个古老的启蒙问题: 给定多项式 $f(x)$, 求出 $f(x)^n$ 的各项系数.
  - 记 $g(x) = f(x)^n$, 那么可以利用 $g'f = nf'g$ 来递推.
  - 如果 $f(x)$ 不是低次多项式, 而是 **稀疏多项式**, 方法仍然奏效, 但难以用 Min25 BM 解决.
  - **稀疏整式递推**, 需要理解操作原理才能得到 **有效** 递推式的问题.
- **隐式整式递推**: 答案序列 $f(n)$ 本身并非整式递推, 但是计算某一项的时候具有整式递推的内核.
  - 截取–Taylor–截取: $\mathcal{O}(n)$ 计算 Bernoulli 数 $B_n = [x^n/n!]\frac{x}{\exp x - 1}$.
  - 思考题: $\tilde{\mathcal{O}}(n)$ 计算 n 个顶点的, **不存在** 2 度点的图的数量.

没有一劳永逸的方法

- 微分有限这一定义本身并不能完整捕捉 $\overset{\text{effective}}{\text{有效}}$ 的递推式.
- 回归一个古老的启蒙问题: 给定多项式 $f(x)$, 求出 $f(x)^n$ 的各项系数.
  - 记 $g(x) = f(x)^n$, 那么可以利用 $g'f = nf'g$ 来递推.
  - 如果 $f(x)$ 不是低次多项式, 而是 **稀疏多项式**, 方法仍然奏效, 但难以用 Min25 BM 解决.
  - **稀疏整式递推**, 需要理解操作原理才能得到 **有效** 递推式的问题.
- **隐式整式递推**: 答案序列 $f(n)$ 本身并非整式递推, 但是计算某一项的时候具有整式递推的内核.
  - 截取–Taylor–截取: $\mathscr{O}(n)$ 计算 Bernoulli 数 $B_n = [x^n/n!]\frac{x}{\exp x - 1}$.
  - 思考题: $\tilde{\mathscr{O}}(n)$ 计算 n 个顶点的, **不存在** 2 度点的图的数量.
- 相信大家的智慧!

感谢倾听

“此时相望不相闻，愿逐月华流照君.”

PinkRabbit sys. ix35 he____he yyc 樱初音 negiizhao
感谢: 陈亮舟 , 任舍予, 史钰申, 万成章, 许庭强 , 杨亦诚 , 赵雨扬 † 协助我准备本次报告.

† 按照字典顺序排列.