形式的冪級数勉強会

2025/10/25

Harui (Twitter: @Nutoriaezu88808, GitHub: @Harui-i)

許して

私は代数をよく知らないこと、体系的にFPSを学んではいないことから、間違いや不適切な表現があるかもしれません。もしあれば、教えていただけると幸いです。

扱わない内容

- 実装の詳細
    - 色々な計算について、定数倍が速いアルゴリズムがあったり、NTTの高速な実装があったり、色々と面白いトピックはあるんですが、複雑になるので省きます。

要求する知識

- 多項式の和/差/積について知っている

FPSとは？

数列 $(a_n)_{n=0}^{\infty}$ に対して、$\sum_{n=0}^{\infty} a_n x^n$ と書くものを通常型母関数(Ordinary Generating Function, **OGF**)といいます。

$\sum_{n=0}^{\infty} a_n \frac{x^n}{n!}$ と書くものを指数型母関数(Exponential Generating Function, **EGF**)といいます。

このように、数列に対応する冪級数を考えている、ということを意識することが重要な気がしています。

例

- 数列 $(3, 1, 4, 1, 5, 9)$に対応するOGFは $3 + 1x + 4x^2 + 1x^3 + 5x^4 + 9x^5$
- 数列 $(3, 1, 4, 1, 5, 9)$に対応するEGFは $3 + 1\frac{x}{1!} + 4\frac{x^2}{2!} + 1\frac{x^3}{3!} + 5\frac{x^4}{4!} + 9\frac{x^5}{5!} = 3 + x + 2x^2 + \frac{x^3}{3!} + \frac{x^4}{4!} + \frac{x^5}{5!}$

いろいろな言葉について

- FPS(Formal Power Series) = 形式的冪級数
- OGF(Ordinary Generating Function) = 通常型母関数
- EGF(Exponential Generating Function) = 指数型母関数

形式的冪級数の演算や性質

ここでは、数列のことは一旦忘れて、FPSの演算や性質について扱います。 なので、数列$\to$FPSの対応付けの違いであるOGF/EGFの違いは一旦忘れても大丈夫です。

FPSの和/差/積/微分/積分は、通常の多項式と同じように定義されます。

和/差/積

$f=\sum_{n=0}^{\infty} a_n x^n, g=\sum_{n=0}^{\infty} b_n x^n$ としたとき、

$$f+g=\sum_{n=0}^{\infty}(a_n+b_n)x^n$$

$$fg=\sum_{n=0}^{\infty}\left(\sum_{j=0}^{n} a_j b_{n-j}\right)x^n$$

微分

極限を考えているわけではなく、こういう操作に微分という名前をつけているだけです。

$$f' = \sum_{n=1}^{\infty} n a_n x^{n-1}$$

逆元

ここからはおそらく新しい演算に見えると思います。
係数は $\mathbb{Q}$ や $\mathbb{F}_{998244353}$ のような体上であるとします。

$f = \sum_{n=0}^{\infty} a_n x^n$ に対して、$g = \sum_{n=0}^{\infty} b_n x^n$ を $fg = 1$ (この右辺の 1 は $(1, 0, 0, \ldots)$ に対応するFPS) を満たすような gを、f の逆元と呼びます。

fの逆元は、$a_0 \neq 0$ なら一意に存在して、そうでないと存在しません。(証明略)(逆元を手計算で計算する方法をしれば、この条件は納得できると思います)

逆元の計算方法・例

$f = 1 - x$ の逆元 $g = \sum_{n=0}^{\infty} b_n x^n$ を求めます。

$fg = 1$ と、
$fg = (1-x)g = g - xg = \sum_{n=0}^{\infty} b_n x^n - \sum_{n=0}^{\infty} b_n x^{n+1}$
$= b_0 + \sum_{n=1}^{\infty} (b_n - b_{n-1}) x^n$ から、$b_0 = 1,\ b_n = b_{n-1} (n \geq 1)$

がわかるので、$g = \sum_{n=0}^{\infty} 1x^n = 1 + x + x^2 + x^3 + \ldots$ となります。

つまり、

$$1 = (1-x)(1 + x + x^2 + x^3 + \ldots)$$

こういう意味で、

$$\frac{1}{1-x} = 1 + x + x^2 + x^3 + \ldots$$

と書いている気がします(要出展)

逆元の手計算

$1-x$の逆元が$1+x+x^2+...$であるのは知っている思いますが、$1-\frac{x^2}{6}+\frac{x^4}{120}$の逆元を手計算で求めるときは、ふつうに筆算をすればよいです(定義とも一致します)。

ここまでの演算について(仮)

ここまでの定義より、**和/差/積/逆元の演算のN項目までは、元のFPSのN項目までの係数から計算できます。**

なので、FPSの定義では無限の項数が出てきますが、実際に計算したいFPSのN項目を求めるときは有限の計算で求められます。

exp

$f = \sum_{n=0}^{\infty} a_n x^n$ (ただし、$a_0 = 0$)に対して、

$$\exp(f) = \sum_{n=0}^{\infty} \frac{f^n}{n!}$$

とします。$a_0 \neq 0$ の場合、$\exp(f)$の0次の項にfの係数の無限和が現れてしまいかなり「気ぃ悪い」ことになります。
(たとえば、FPSの性質である和/差/積のn次目までの係数が、n次目までの係数だけで決まる、という性質が崩れます)

expの性質

- $\exp(f+g) = \exp(f)\exp(g)$
- $(\exp(f))' = f'\exp(f)$

log

$f = \sum_{n=0}^{\infty} a_n x^n$ (ただし、$a_0 = 1$)に対して、

$$\log(f) = \sum_{n=1}^{\infty} (-1)^{n-1} \frac{(f-1)^n}{n}$$

とします。$a_0 \neq 1$ の場合、$\log(f)$の0次の項にfの係数の無限和が現れてしまいかなり「気ぃ悪い」ことになります。
(たとえば、FPS性質である和/差/積のn次目までの係数が、n次目までの係数だけで決まる、という性質が崩れます)

logの性質

TODO: 書く

- $(\log f)' = \dfrac{f'}{f}$
- $\log(fg) = \log f + \log g$

微分の嬉しい性質(OGF)

$f = \sum_{n=0}^{\infty} a_n x^n$ のとき、

$$xf' = \sum_{n=0}^{\infty} n a_n x^n$$

つまり、OGFにおいて、微分してxをかける操作は、数列の各項にそのインデックスをかける操作に対応します。

微分の嬉しい性質(EGF)

$f = \sum_{n=0}^{\infty} a_n \frac{x^n}{n!}$ のとき、

$$f' = \sum_{n=0}^{\infty} a_{n+1} \frac{x^n}{n!}$$

つまり、EGFにおいて、微分する操作は、数列の各項を一つ前にずらす操作に対応します。

累積和(OGF)

$f = \sum_{n=0}^{\infty} a_n x^n$ のとき、

$$\frac{f}{1-x} = \sum_{n=0}^{\infty} \left(\sum_{m=0}^{n} a_m \right) x^n$$

つまり、OGFにおいて、$\frac{1}{1-x}$ をかける操作は、数列の累積和をとる操作に対応します。

OGFの積

(あたりまえかも)

$f = \sum_{n=0}^{\infty} a_n x^n, g = \sum_{n=0}^{\infty} b_n x^n$ のとき、

$$fg = \sum_{n=0}^{\infty} \left(\sum_{m=0}^{n} a_m b_{n-m} \right) x^n$$

OGFの積(続き)

$f=\sum_{n=0}^{\infty} a_n x^n, g=\sum_{n=0}^{\infty} b_n x^n, h=\sum_{n=0}^{\infty} c_n x^n$ としたとき

$$fgh=\sum_{n=0}^{\infty} x^n \sum_{n_1+n_2+n_3=n} a_{n_1} b_{n_2} c_{n_3}$$

もっと一般に3個以上のOGFの積についても同様で、x^nの係数には、次数和がnであるような各項の選び型についての係数の積の総和が現れる。

OGFの積の例(重複組み合わせ, 負の二項定理)

数列 $(1,1,1,\dots)$ に対応するOGFは

$$\frac{1}{1-x} = 1 + x + x^2 + \cdots$$

でした。これをK個掛け合わせると、

$$\left(\frac{1}{1-x}\right)^K = (1 + x + x^2 + \cdots)^K = \sum_{n=0}^{\infty}\left(\sum_{n_1+\cdots+n_K=n} 1\right)x^n$$

となり、x^nの係数は「nをK個の非負整数の和として書く方法の数(足す順番の区別あり)」すなわち重複組合せの数に一致します。
(n個の玉と$K-1$個の仕切りを並べる方法の数え上げ)

OGFの積の例(重複組み合わせ, 負の二項定理 続き)

$$\left(\frac{1}{1-x}\right)^K = \sum_{n=0}^{\infty} \binom{n+K-1}{K-1} x^n$$

これを負の二項定理と呼んだりすることもあるらしいです。

OGFの積の例(重複組み合わせ, 負の二項定理 まとめ)

$$\sum_{n_1+n_2+...+n_K=n} 1 = \binom{n+K-1}{K-1}$$

これは、「n個の玉と$K-1$個の仕切りの並べかえの数」です。

写像12相的に言うと **「n個の区別しない玉をK個の区別する箱に入れる場合の数(箱に入れる数は0でもいい)」** でもあります。

EGFの積

$f = \sum_{n=0}^{\infty} a_n \frac{x^n}{n!}, g = \sum_{n=0}^{\infty} b_n \frac{x^n}{n!}$ のとき、

$$fg = \sum_{n=0}^{\infty} \left(\sum_{j=0}^{n} \binom{n}{j} a_j b_{n-j} \right) \frac{x^n}{n!}$$

つまり、$(c_n)_{n=0}^{\infty}$の母関数がfgであるとき、$c_n = \sum_{j=0}^{n} \binom{n}{j} a_j b_{n-j}$ となります。

OGFの場合との違いは、係数の積に二項係数の重みがついていることです。

EGFの積(続き)

$f = \sum_{n=0}^{\infty} a_n \frac{x^n}{n!}, g = \sum_{n=0}^{\infty} b_n \frac{x^n}{n!}, h = \sum_{n=0}^{\infty} c_n \frac{x^n}{n!}$ としたとき

$$fgh = \sum_{n=0}^{\infty} \frac{x^n}{n!} \sum_{n_1+n_2+n_3=n} \frac{n!}{n_1!n_2!n_3!} a_{n_1} b_{n_2} c_{n_3}$$

となります。4個以上のときも同様で、EGFの積の数列のn項目には、和がnとなるような次数の選び方について、係数の積に多項係数で重み付けしたものの総和が現れます。

EGFの積の例(指数関数の冪)

数列 $(1, 1, 1, \ldots)$ に対応するEGFは e^x です。これをK個掛け合わせると

$$(e^x)^K = e^{Kx} = \sum_{n=0}^{\infty} \frac{(Kx)^n}{n!} = \sum_{n=0}^{\infty} K^n \frac{x^n}{n!}$$

となります。EGFの積を考えると、以下のように捉えることもできます。

$$K^n \frac{x^n}{n!} = \frac{x^n}{n!} \sum_{n_1+n_2+\ldots+n_K=n} \frac{n!}{n_1!n_2!\ldots n_K!} a_{n_1} b_{n_2} \ldots c_{n_K}$$

つまり、

$$\sum_{n_1+n_2+\ldots+n_K=n} \frac{n!}{n_1!n_2!\ldots n_K!} = K^n$$

です。これは、n個の区別する玉をK個の区別する箱に入れる場合の数です。

OGFの積の場合

$$\sum_{n_1+n_2+\ldots+n_K=n} 1 = \binom{n+K-1}{K-1}$$

- これは、n個の**区別しない玉**をK個の区別する箱に入れる場合の数

EGFの積の場合(指数関数の冪 まとめ)

$$\sum_{n_1+n_2+\ldots+n_K=n} \frac{n!}{n_1!n_2!\ldots n_K!} = K^n$$

- これは、n個の**区別する玉**をK個の区別する箱に入れる場合の数

このように、区別する/しないの違いがOGF/EGFの違いと対応することもあります(他の例もあるかも)

FPSを使って問題を解く

ABC 178 D - Redistribution

> 整数 S が与えられます。すべての項が3以上の整数で、その総和がSであるような数列がいくつあるか求めてください $\pmod{10^9+7}$

$f = x^3 + x^4 + x^5 + \ldots$ とすると、$f + f^2 + f^3 + \ldots$ の x^S の係数が答え。

$$1 + f + f^2 + f^3 + \ldots = \frac{1}{1-f},\ f = \frac{x^3}{1-x}$$

なので、「x^3に$1-x$の逆元をかけて、マイナス1倍してそれに1を足したものの逆元」を計算すれば求められる。
(実は $\frac{1}{1-f} = \frac{1}{1-\frac{x^3}{1-x}} = \frac{1-x}{1-x-x^3}$ なので、もっと簡単になる。母関数が有理関数なので線形漸化式があり、それを計算すればもっと単純になる)

積の和典型(続き)

> 長さがN ,各要素が1以上M以下の整数列 $(a_1, a_2, ..., a_N)$ であって、和がMであるような全ての数列について$\prod_i a_i$を求めて、その総和求めよ。

$f^N = (x(\frac{1}{1-x})')^N = (x(1-x)^{-2})^N = (\frac{x}{(1-x)^2})^N$ であり、 実は $(1-x)^{-K} = \sum_{n=0}^{\infty} \binom{n+K-1}{K-1} x^n$ なので、二項係数さえもとまっていれば、f^NのM次の係数は$\Theta(1)$ で計算できる。

$(1-x)^{-K}$のn次の係数が二項係数で表せる理由: 区別するK個の箱に、区別しないn個の玉を入れる方法の数え上げ(玉と仕切りを考えるやつ)なので、K個の玉と$n-1$個の仕切りを並べる方法の数え上げに対応するから。

積の和典型(続き)

積の和典型あるあるとして「掛けたいものを係数に持ってきて、和を取りたいものを指数に持ってきがち」というのがある

積の和典型の例

ABC 231 G - Balls in Boxes

> 1からNの番号がついたN個の箱があります。各箱にはそれぞれA_i個のボールが入っています。 あなたは、次の操作をK回行います。
>
> - 1からNの間で整数iを等確率で選び、箱iにボールを1個追加する。
>
> K回の操作が終了した後で箱iに入っているボールの個数をB_iとするとき、**スコア**はボールの個数の総積$\prod_{i=1}^{N} B_i$で定義されます。 スコアの期待値を求めてください。 $\pmod{998244353}$

注意: 次のスライドにネタバレあり

ABC 231 G - Balls in Boxes(続き)

期待値を求めているけど、等確率で選んでいるから、全ての場合についてスコアの和を求めて最後にN^Kで割れば期待値も求まる。

そう考えると、積の和典型っぽい。 箱iに追加するボールの個数をa_iとすると、最終的なスコアは $\prod_{i=1}^{N}(A_i + a_i)$ であり、 $\sum_{i=1}^{N} a_i = K$ である。 今回の問題では「箱iに追加するボールの個数」が積の和典型でいう「数列の各要素」」に対応していそう。 今回の問題の「ボールの個数の合計」が積の和典型でいう「数列の和」に対応していそう。

となると、となると、 $f_i = A_i + (A_i + 1)x + (A_i + 2)x^2 + (A_i + 3)x^3 + \dots$ として、 $\prod_{i=1}^{N} f_i$ の x^K の係数を求めればよさそう。

ABC 231 G - Balls in Boxes(続き)

実際にその方針で$N=2, K=2$で計算してみる

$$f_1 = A_1 + (A_1+1)x + (A_1+2)x^2 + \dots$$
$$f_2 = A_2 + (A_2+1)x + (A_2+2)x^2 + \dots$$

x^2の係数は $A_1(A_2+2)+(A_1+1)(A_2+1)+(A_1+2)A_2$となる。

- $A_1(A_2+2)$の項は、箱1に0個、箱2に2個追加する場合に対応している。
  - あってそう
- $(A_1+1)(A_2+1)$は箱1に1個、箱2に1個追加する場合に対応しているしている
  - 間違っている。箱1に先に入れる場合とあとに入れる場合があるので、2倍しないと合わない。
- $(A_1+2)A_2$は箱1に2個、箱2に0個追加する場合に対応している。
  - あってそう

ABC 231 G - Balls in Boxes(続き)

Q: なぜズレたか？？

A: 箱1に3個、箱2に1個、箱3にに4個追加する場合の数は $\frac{(3+1+4)!}{3!1!4!}$ になる。このように多項係数の重みをつけなければいけなかったから。

そう考えると、$f_i = A_i + \frac{(A_1+1)}{1!}x + \frac{(A_i+2)}{2!}x^2 + ...$ として、$\prod_{i=1}^{N} f_i$ の x^K の係数を求めればよさそう。 (先に学んだとおり、EGFの積には係数の重みに多項係数がつくのでので)

$f_i = (A_i + x)e^x$なので、$\prod_{i=1}^{N} f_i = (\prod_{i=1}^{N}(A_i + x))e^{Nx}$ となる。 $\prod_{i=1}^{N}(A_i + x)$はN次多項式なので、あとは展開すればよい。 (e^{Nx}のn次の項は $\frac{(Nx)^n}{n!}$ なので、x^Kの係数を求めるときはN次多項式の$K - n$次の項を見ればよい)

参考文献

- https://codeforces.com/blog/entry/77468
  - OGF/EGFの定義からいろいろな性質まで扱っている(英語)
- https://trap.jp/post/1657/
  - traPのFPSの講習会の資料。計算方法や、問題を手計算で解く方法などが詳しく書かれている。
- https://maspypy.com/多項式・形式的べき級数数え上げとの対応付け
  - FPSで数え上げ問題を解く方法について。 (2)や(3)も参考になる。