倍增思想的一个简单应用

blutrex

2017/03/12

问题描述

下面这道题多次运用倍增思想，这个算法毕姥爷多次提到过。

下面这道题多次运用倍增思想，这个算法毕姥爷多次提到过。大佬~~wxh010910~~肯定看一眼就秒了。

问题描述

下面这道题多次运用倍增思想，这个算法毕姥爷多次提到过。
大佬~~wxh010910~~肯定看一眼就秒了。
我太弱了，只会非常基础的多项式运算。

问题描述

下面这道题多次运用倍增思想，这个算法毕姥爷多次提到过。
大佬~~wxh010910~~肯定看一眼就秒了。
我太弱了，只会非常基础的多项式运算。
~~更复杂的应用请下一位zyz讲~~。

问题描述

本题时限4 s。

递推数列II

给出长为m的数列$\boldsymbol{a} = \langle a_0, a_1, \ldots, a_{m-1} \rangle$，以及无穷数列$\boldsymbol{f}$的前$m$项$\langle f_0, f_1, \ldots, f_{m-1} \rangle$，对于$\forall i \geq m$，都有

$$f_i = \sum_{j=0}^{m-1} a_j f_{i-j-1}$$

求$\boldsymbol{f}$的第n项f_{n-1}，输出答案对1811939329取模的结果。
$1 \leq m \leq 30000$，$1 \leq n \leq 10^{18}$，$0 \leq a_j, f_i < 1811939329$。

问题描述

本题时限4 s。

递推数列II

给出长为m的数列$\boldsymbol{a} = \langle a_0, a_1, \ldots, a_{m-1}\rangle$，以及无穷数列$\boldsymbol{f}$的前$m$项$\langle f_0, f_1, \ldots, f_{m-1}\rangle$，对于$\forall i \geq m$，都有

$$f_i = \sum_{j=0}^{m-1} a_j f_{i-j-1}$$

求$\boldsymbol{f}$的第n项f_{n-1}，输出答案对1811939329取模的结果。
$1 \leq m \leq 30000$，$1 \leq n \leq 10^{18}$，$0 \leq a_j, f_i < 1811939329$。

传统做法是$\Theta(m^3 \log n)$的矩阵快速幂，复杂度太高。

算法I

有一个$\Theta(m^2 \log n)$的倍增算法，适用于模数为一般质数的情况（BZOJ4161）。

算法I

有一个$\Theta(m^2 \log n)$的倍增算法，适用于模数为一般质数的情况（BZOJ4161）。
设数列$\boldsymbol{a}^{[l]}$满足对于$\forall i \geq m + l - 1$

$$f_i = \sum_{j=0}^{m-1} a_j^{[l]} f_{i-j-l}$$

算法I

有一个$\Theta(m^2 \log n)$的倍增算法，适用于模数为一般质数的情况（BZOJ4161）。
设数列$\boldsymbol{a}^{[l]}$满足对于$\forall i \geq m+l-1$

$$f_i = \sum_{j=0}^{m-1} a_j^{[l]} f_{i-j-l}$$

显然对于任意正整数l，这样的$\boldsymbol{a}^{[l]}$总存在，其中$\boldsymbol{a}^{[1]} = \boldsymbol{a}$。

算法I

有一个$\Theta(m^2\log n)$的倍增算法，适用于模数为一般质数的情况（BZOJ4161）。
设数列$\boldsymbol{a}^{[l]}$满足对于$\forall i\geq m+l-1$

$$f_i=\sum_{j=0}^{m-1}a_j^{[l]}f_{i-j-l}$$

显然对于任意正整数l，这样的$\boldsymbol{a}^{[l]}$总存在，其中$\boldsymbol{a}^{[1]}=\boldsymbol{a}$。
则答案为

$$f_{n-1}=\sum_{j=0}^{m-1}a_j^{[n-m]}f_{m-j-1}$$

算法I

有一个$\Theta(m^2\log n)$的倍增算法，适用于模数为一般质数的情况（BZOJ4161）。
设数列$\boldsymbol{a}^{[l]}$满足对于$\forall i \geq m+l-1$

$$f_i = \sum_{j=0}^{m-1} a_j^{[l]} f_{i-j-l}$$

显然对于任意正整数l，这样的$\boldsymbol{a}^{[l]}$总存在，其中$\boldsymbol{a}^{[1]} = \boldsymbol{a}$。
则答案为

$$f_{n-1} = \sum_{j=0}^{m-1} a_j^{[n-m]} f_{m-j-1}$$

问题转化为如何构造出$\boldsymbol{a}^{[n-m]}$。

算法I

如果构造出了$\boldsymbol{a}^{[k]}$和$\boldsymbol{a}^{[l]}$，就可以构造出$\boldsymbol{a}^{[k+l]}$。

算法I

如果构造出了$\boldsymbol{a}^{[k]}$和$\boldsymbol{a}^{[l]}$，就可以构造出$\boldsymbol{a}^{[k+l]}$。

$$\begin{aligned} f_i &= \sum_{j^{[k]}=0}^{m-1} a^{[k]}_{j^{[k]}} f_{i-j^{[k]}-k} \\ &= \sum_{j^{[k]}=0}^{m-1} a^{[k]}_{j^{[k]}} \sum_{j^{[l]}=0}^{m-1} a^{[l]}_{j^{[l]}} f_{i-j^{[k]}-j^{[l]}-k-l} \\ &= \sum_{j=0}^{2m-2} b_j f_{i-j-k-l} \end{aligned}$$

其中

$$b_j = \sum_{j^{[k]}+j^{[l]}=j} a^{[k]}_{j^{[k]}} a^{[l]}_{j^{[l]}}$$

算法I

但是$\boldsymbol{b}$的长度为$2m-1$，超过了m，可以发现其中一些是可以用$\boldsymbol{a}$表示出来的，我们需要将这些项去掉。

算法I

但是$\boldsymbol{b}$的长度为$2m-1$，超过了m，可以发现其中一些是可以用$\boldsymbol{a}$表示出来的，我们需要将这些项去掉。

可以每次将$\boldsymbol{b}$的长度减1。设当前$\boldsymbol{b}$的长度为$c\ (c > m)$。

算法I

但是$\boldsymbol{b}$的长度为$2m-1$，超过了m，可以发现其中一些是可以用$\boldsymbol{a}$表示出来的，我们需要将这些项去掉。
可以每次将$\boldsymbol{b}$的长度减1。设当前$\boldsymbol{b}$的长度为$c\ (c>m)$。
令$q=b_{c-1}/a_{m-1}$，注意到1811939329是质数，只要$a_{m-1}\neq 0$则逆元一定存在，否则可以将m变小直到$m=0$或$a_{m-1}\neq 0$。

$$
\begin{aligned}
b_{c-m-1} &:= b_{c-m-1}+q \\
b_{c-j-1} &:= b_{c-j-1}-qa_j\ (0\leq j<m)
\end{aligned}
$$

算法I

但是$\boldsymbol{b}$的长度为$2m-1$，超过了m，可以发现其中一些是可以用$\boldsymbol{a}$表示出来的，我们需要将这些项去掉。
可以每次将$\boldsymbol{b}$的长度减1。设当前$\boldsymbol{b}$的长度为c $(c>m)$。
令$q=b_{c-1}/a_{m-1}$，注意到1811939329是质数，只要$a_{m-1}\neq 0$则逆元一定存在，否则可以将m变小直到$m=0$或$a_{m-1}\neq 0$。

$$b_{c-m-1} := b_{c-m-1}+q$$
$$b_{c-j-1} := b_{c-j-1}-qa_j \ (0\le j<m)$$

直到$c=m$时，$\boldsymbol{a}^{[k+l]}=\boldsymbol{b}$。这样就在$\Theta(m^2)$的时间内求出了$\boldsymbol{a}^{[k+l]}$。

算法I

但是$\boldsymbol{b}$的长度为$2m-1$，超过了m，可以发现其中一些是可以用$\boldsymbol{a}$表示出来的，我们需要将这些项去掉。
可以每次将$\boldsymbol{b}$的长度减1。设当前$\boldsymbol{b}$的长度为c $(c>m)$。
令$q=b_{c-1}/a_{m-1}$，注意到1811939329是质数，只要$a_{m-1}\neq 0$则逆元一定存在，否则可以将m变小直到$m=0$或$a_{m-1}\neq 0$。

$$b_{c-m-1} := b_{c-m-1}+q$$
$$b_{c-j-1} := b_{c-j-1}-qa_j \ (0\le j<m)$$

直到$c=m$时，$\boldsymbol{a}^{[k+l]}=\boldsymbol{b}$。这样就在$\Theta(m^2)$的时间内求出了$\boldsymbol{a}^{[k+l]}$。
要求出$\boldsymbol{a}^{[n-m]}$，可以倍增构造，由$\boldsymbol{a}^{[l]}$求出$\boldsymbol{a}^{[l+1]}$或$\boldsymbol{a}^{[2l]}$，这样只会进行$\Theta(\log n)$次，总复杂度是$\Theta(m^2\log n)$。

注意到模数$1811939329 = 2^{26} \times 3^3 + 1$，可以进行NTT。

注意到模数$1811939329 = 2^{26} \times 3^3 + 1$，可以进行NTT。
注意到求$\boldsymbol{b}$相当于卷积，最后维护$\boldsymbol{b}$的过程类似多项式除法，所以可以用多项式相关算法优化到$\Theta(m \log m \log n)$。

算法II

注意到模数$1811939329 = 2^{26} \times 3^3 + 1$，可以进行NTT。
注意到求$\boldsymbol{b}$相当于卷积，最后维护$\boldsymbol{b}$的过程类似多项式除法，所以可以用多项式相关算法优化到$\Theta(m \log m \log n)$。
下面先讲一下多项式的基本运算。

前置技能

多项式

前置技能

多项式

- 乘法

前置技能

多项式

- 乘法
- 求逆

前置技能

多项式

- 乘法
- 求逆
- 除法

符号约定

设有多项式$F(x)=\sum_{k=0}^{n-1} f_k x^k$。

符号约定

设有多项式$F(x) = \sum_{k=0}^{n-1} f_k x^k$。
定义其次数界$\deg F = n$。

符号约定

设有多项式$F(x)=\sum_{k=0}^{n-1} f_k x^k$。
定义其次数界$\deg F=n$。
定义f_k为$F(x)$的k次项系数。

符号约定

设有多项式$F(x) = \sum_{k=0}^{n-1} f_k x^k$。
定义其次数界$\deg F = n$。
定义f_k为$F(x)$的k次项系数。
定义其反多项式为$F^{\mathrm{R}}(x) = x^{n-1} F(x^{-1}) = \sum_{k=0}^{n-1} f_{n-k-1} x^k$。

符号约定

设有多项式$F(x)=\sum_{k=0}^{n-1} f_k x^k$。
定义其次数界$\deg F=n$。
定义f_k为$F(x)$的k次项系数。
定义其反多项式为$F^{\mathrm{R}}(x)=x^{n-1}F(x^{-1})=\sum_{k=0}^{n-1} f_{n-k-1}x^k$。
下文中，如不指明，则同一个字母的大小写表示对应的多项式和系数。

直接FFT就好，各位dalao一定非常熟练。

多项式求逆

多项式求逆在多项式除法时会用到，在前几天的交流中已被多次提到。

描述

给出多项式$A(x)$，求一个多项式$A^{-1}(x)$满足

$$A(x)A^{-1}(x) \equiv 1 \pmod{x^n}$$

保证$a_0 \neq 0$。这里的n不一定是2的整数次幂。

多项式求逆

我们仍然采用倍增构造。

多项式求逆

我们仍然采用倍增构造。
设$B_n(x)$表示模x^n时的答案。

多项式求逆

我们仍然采用倍增构造。
设$B_n(x)$表示模x^n时的答案。
首先$B_1(x) = a_0^{-1}$。如果$a_0 = 0$，则$A(x)$不存在逆元。

多项式求逆

我们仍然采用倍增构造。
设$B_n(x)$表示模x^n时的答案。
首先$B_1(x) = a_0^{-1}$。如果$a_0 = 0$，则$A(x)$不存在逆元。
假设已经求出了$B_{\lceil \frac{n}{2} \rceil}(x)$，我们要构造$B_n(x)$。

多项式求逆

我们仍然采用倍增构造。
设$B_n(x)$表示模x^n时的答案。
首先$B_1(x) = a_0^{-1}$。如果$a_0 = 0$，则$A(x)$不存在逆元。
假设已经求出了$B_{\lceil \frac{n}{2} \rceil}(x)$，我们要构造$B_n(x)$。
因为

$$A(x)B_{\lceil \frac{n}{2} \rceil}(x) \equiv 1 \pmod{x^{\lceil \frac{n}{2} \rceil}}$$
$$A(x)B_n(x) \equiv 1 \pmod{x^{\lceil \frac{n}{2} \rceil}}$$

多项式求逆

我们仍然采用倍增构造。
设$B_n(x)$表示模x^n时的答案。
首先$B_1(x) = a_0^{-1}$。如果$a_0 = 0$，则$A(x)$不存在逆元。
假设已经求出了$B_{\lceil \frac{n}{2} \rceil}(x)$，我们要构造$B_n(x)$。
因为

$$A(x)B_{\lceil \frac{n}{2} \rceil}(x) \equiv 1 \pmod{x^{\lceil \frac{n}{2} \rceil}}$$
$$A(x)B_n(x) \equiv 1 \pmod{x^{\lceil \frac{n}{2} \rceil}}$$

两式相减，得

$$A(x)(B_n(x) - B_{\lceil \frac{n}{2} \rceil}(x)) \equiv 0 \pmod{x^{\lceil \frac{n}{2} \rceil}}$$

多项式求逆

因为

$$A(x) \not\equiv 0 \pmod{x^{\lceil \frac{n}{2} \rceil}}$$

多项式求逆

因为

$$A(x) \not\equiv 0 \pmod{x^{\lceil \frac{n}{2} \rceil}}$$

所以

$$B_n(x) - B_{\lceil \frac{n}{2} \rceil}(x) \equiv 0 \pmod{x^{\lceil \frac{n}{2} \rceil}}$$

多项式求逆

因为

$$A(x) \not\equiv 0 \pmod{x^{\lceil \frac{n}{2} \rceil}}$$

所以

$$B_n(x) - B_{\lceil \frac{n}{2} \rceil}(x) \equiv 0 \pmod{x^{\lceil \frac{n}{2} \rceil}}$$

平方，得

$$B_n^2(x) + B_{\lceil \frac{n}{2} \rceil}^2(x) - 2B_n(x)B_{\lceil \frac{n}{2} \rceil}(x) \equiv 0 \pmod{x^n}$$

多项式求逆

因为

$$A(x) \not\equiv 0 \pmod{x^{\lceil \frac{n}{2} \rceil}}$$

所以

$$B_n(x) - B_{\lceil \frac{n}{2} \rceil}(x) \equiv 0 \pmod{x^{\lceil \frac{n}{2} \rceil}}$$

平方，得

$$B_n^2(x) + B_{\lceil \frac{n}{2} \rceil}^2(x) - 2B_n(x)B_{\lceil \frac{n}{2} \rceil}(x) \equiv 0 \pmod{x^n}$$

两边乘$A(x)$并整理，得

$$B_n(x) \equiv B_{\lceil \frac{n}{2} \rceil}(x)(2 - A(x)B_{\lceil \frac{n}{2} \rceil}(x)) \pmod{x^n}$$

多项式求逆

因为

$$A(x) \not\equiv 0 \pmod{x^{\lceil \frac{n}{2} \rceil}}$$

所以

$$B_n(x) - B_{\lceil \frac{n}{2} \rceil}(x) \equiv 0 \pmod{x^{\lceil \frac{n}{2} \rceil}}$$

平方，得

$$B_n^2(x) + B_{\lceil \frac{n}{2} \rceil}^2(x) - 2B_n(x)B_{\lceil \frac{n}{2} \rceil}(x) \equiv 0 \pmod{x^n}$$

两边乘$A(x)$并整理，得

$$B_n(x) \equiv B_{\lceil \frac{n}{2} \rceil}(x)(2 - A(x)B_{\lceil \frac{n}{2} \rceil}(x)) \pmod{x^n}$$

时间复杂度为$T(n) = T(n/2) + \Theta(n \log n) = \Theta(n \log n)$。
~~常数巨大。~~

多项式除法

描述

给出次数界为m的多项式$A(x)$和次数界为n的多项式$B(x)$，求两个多项式$Q(x)$，$R(x)$，满足$R(x)$的次数界小于n，且

$$A(x) = B(x)Q(x) + R(x)$$

保证$b_{n-1} \neq 0$。

多项式除法

同时有$Q(x)$和$R(x)$难以处理，考虑消去$R(x)$的影响。

多项式除法

同时有$Q(x)$和$R(x)$难以处理，考虑消去$R(x)$的影响。
将x变为x^{-1}，再乘x^{m-1}，得

$$x^{m-1}A(x^{-1}) = x^{n-1}B(x^{-1})x^{m-n}Q(x^{-1}) + x^{m-1}R(x^{-1})$$

同时有$Q(x)$和$R(x)$难以处理，考虑消去$R(x)$的影响。
将x变为x^{-1}，再乘x^{m-1}，得

$$x^{m-1}A(x^{-1}) = x^{n-1}B(x^{-1})x^{m-n}Q(x^{-1}) + x^{m-1}R(x^{-1})$$

即

$$A^{\mathrm{R}}(x) = B^{\mathrm{R}}(x)Q^{\mathrm{R}}(x) + x^{m-n+1}R^{\mathrm{R}}(x)$$

多项式除法

多项式除法

放到模x^{m-n+1}下，$R(x)$就被消去了。因为$\deg Q = m - n + 1$，所以$Q(x)$不受影响。

$$A^{\mathrm{R}}(x) \equiv B^{\mathrm{R}}(x) Q^{\mathrm{R}}(x) \pmod{x^{m-n+1}}$$

多项式除法

放到模x^{m-n+1}下，$R(x)$就被消去了。因为$\deg Q = m - n + 1$，所以$Q(x)$不受影响。

$$A^{\mathrm{R}}(x) \equiv B^{\mathrm{R}}(x) Q^{\mathrm{R}}(x) \pmod{x^{m-n+1}}$$

即

$$Q^{\mathrm{R}}(x) \equiv A^{\mathrm{R}}(x) B^{\mathrm{R}^{-1}}(x) \pmod{x^{m-n+1}}$$

多项式除法

放到模x^{m-n+1}下，$R(x)$就被消去了。因为$\deg Q = m - n + 1$，所以$Q(x)$不受影响。

$$A^{\mathrm{R}}(x) \equiv B^{\mathrm{R}}(x) Q^{\mathrm{R}}(x) \pmod{x^{m-n+1}}$$

即

$$Q^{\mathrm{R}}(x) \equiv A^{\mathrm{R}}(x) B^{\mathrm{R}^{-1}}(x) \pmod{x^{m-n+1}}$$

这样就可以用多项式求逆求出$Q(x)$。

多项式除法

放到模x^{m-n+1}下，$R(x)$就被消去了。因为$\deg Q = m - n + 1$，所以$Q(x)$不受影响。

$$A^{\mathrm{R}}(x) \equiv B^{\mathrm{R}}(x)Q^{\mathrm{R}}(x) \pmod{x^{m-n+1}}$$

即

$$Q^{\mathrm{R}}(x) \equiv A^{\mathrm{R}}(x){B^{\mathrm{R}}}^{-1}(x) \pmod{x^{m-n+1}}$$

这样就可以用多项式求逆求出$Q(x)$。
$R(x) = A(x) - B(x)Q(x)$也可以求出。

放到模x^{m-n+1}下，$R(x)$就被消去了。因为$\deg Q = m - n + 1$，所以$Q(x)$不受影响。

$$A^{\mathrm{R}}(x) \equiv B^{\mathrm{R}}(x) Q^{\mathrm{R}}(x) \pmod{x^{m-n+1}}$$

即

$$Q^{\mathrm{R}}(x) \equiv A^{\mathrm{R}}(x) {B^{\mathrm{R}}}^{-1}(x) \pmod{x^{m-n+1}}$$

这样就可以用多项式求逆求出$Q(x)$。
$R(x) = A(x) - B(x)Q(x)$也可以求出。
总复杂度为$\Theta(n \log n)$。

算法II

在算法I中处理$\boldsymbol{b}$的那一步，就相当于是求出$B(x)$除以$A(x)$的商$Q(x)$，再让$q_0 := 0$，然后求出

$$A^{[k+l]}(x) = B(x) - A(x)Q(x) + x^{-1}Q(x)$$

算法II

在算法I中处理$\boldsymbol{b}$的那一步，就相当于是求出$B(x)$除以$A(x)$的商$Q(x)$，再让$q_0 := 0$，然后求出

$$A^{[k+l]}(x) = B(x) - A(x)Q(x) + x^{-1}Q(x)$$

这样复杂度就优化到了$\Theta(m \log m \log n)$。

小结

倍增是一种常见思想，本题中求$\boldsymbol{a}^{[l]}$和多项式求逆都是倍增构造。

小结

倍增是一种常见思想，本题中求$\boldsymbol{a}^{[l]}$和多项式求逆都是倍增构造。

一些常见算法（如后缀数组的简单实现、多项式牛顿迭代等）也是倍增构造。

小结

倍增是一种常见思想，本题中求$\boldsymbol{a}^{[l]}$和多项式求逆都是倍增构造。

一些常见算法（如后缀数组的简单实现、多项式牛顿迭代等）也是倍增构造。

善用倍增也许能起到意想不到的效果。

谢谢大家！