# 解决多项式乘法以及模型：

## 前言：

一般NTT和FFT都是解决卷积问题的算法，理论上都能能用但是代码效率等存在差异：

* 在大数运算范围内，FFT在高精度乘法上的误差分析有人做过，结论就是完全可以忽略，原因我没查过。
* 在大数运算范围内，FFT比NTT稍微快一点，虽然FFT是浮点数运算，但总运算次数比NTT少，NTT要频繁取模。
* 大数里，不论FFT还是NTT都之能压位2位，FFT要满足：

HEX\*len\*len<double极限。NTT要满足：HEX\*len\*len<大素数P

## 问题模型：

A数组是a1 a2 a3 a4 a5….an

B数组是b1 b2 b3 b4 b5….bm

它们代表多项式系数

那么乘积是：

a1\*b1+a1\*b2+ a1\*b3+…+a1\*bm

+ + +

a2\*b1+ a2\*b2+ a2\*b3 +…+a2\*bm

+ + +

A3\*b1+ a3\*b2+ a3\*b3 +…+a3\*bm

……………………….

…………………..

…………………….

结果数组设为x数组

这个也适用于大整数乘法

它等价于：

把A数组的NTT或者FFT得到F1

把B数组的NTT或者FFT得到F2

用F1和F2每一项相乘 加和得到F3

F3[i]=

把F3数组逆NTT或者逆FFT得到x数组

X数组等于上面n\*n复杂度的x结果数组

FFT或者NTT时候应该统一长度L

L一定要大于A B

x每一个长度，是2整数次方

L=比len(A)+len(B)-1大的第一个2的整数次方

因为原本两个多项式乘法结果就应该是len(A)+len(B)-1个数

预处理2的整数次方

int p2[30]={1,2,4,8,16,32,64,128,256,512,1024,2048,4096,8192,16384,32768,65536,131072,262144,524288,1048576,2097152};

A数组B数组补0到L

最后结果的x数组也是L长度的，但是我们只需要输出到

len(A)+len(B)-1

## FFT/NTT的性质：

多项式乘法模型，就是所谓卷积。

另外对于多项式数组xa和xb；设changeN(x)代表转变成比x大的第一个2的次方数. 设它们傅里叶变换后数组是fa,fb,有如下性质：

Xa\*xb=)

Xa+xb=)

加减法规律仅仅只相加之后，变回原本，没有时间优势，但在后求逆里有应用

## 大整数乘法/多项式乘法：

二者很类似

属于同一的模型大整数乘法就是多项式乘法，但是注意，如果取P=998244353, 应该满足HEX\*HEX\*len<P HEX是大整数乘法的进制大小，len是要被NTT数组长度。不满足就会出错，为什么呢？因为NTT一直是在源根g的多少次方模P下的，从来不是相乘结果总体模P，你要保证数组累加后，再算乘法时候两个两个乘不会爆P；

## 多项式求逆：

洛谷4238

deg代表多项式最高次数，称之为多项式的度

n代表被除数多项式有几个项，我们这里说的多项式不忽略系数是0的项，比如x^2,有3项，它相当于ax^2+bx^1+c

对于一个多项式 A(x)，如果存在 B(x)满足 degB≤degA并且

A(x)B(x)≡1(modx^n)

那么称 B(x)为 A(x)在 % x^n  意义下的**逆元**（inverse element），记作 (x)

现在考虑如何求(x)，

当 n=1 时，A(x)%x=c，c是常数，原因是n=1时多项式B只有一个常数项c。那么(x)= 多项式有没有逆元也取决于c有木有逆元

n>1时，

A(x)B(x)≡1(modx^n)

假设在 % x^(n/2) 意义下 A(x) 的逆元是 B1(x) 并且我们已经求出，那么

A(x)(x)≡1(modx^n/2)

我们要求的式子正式：A(x)B (x)≡1(modx^n/2)

那么B(x)=(x) B(x)- (x)=0

两边平方，B(x)+A(x) (x)−2B′(x)≡0(mod xn)

同时乘A(x),由于:A(x)B(x)≡1(modx^n)

**结果是B(x)≡B′(x)(2−A(x)B′(x))(mod x^n)**

**= 2B′(x)−A(x)B′(x) B′(x)(mod x^n)**

**其中B′(x)的项数len(B′(x))= len(B(x))/2，A要用到的项数，等于B’的项数**

用FFT/NTT优化乘法

这个算法时间复杂度是logn\*n\*logn

网上说是n\*;logn的，但其实要做多次FFT/NTT运算，而且NTT/FFT写得好与坏，常数差距太大。要不断优化模板

注意求FFT/NTT运算次数可以优化

## 多项式 除法和取模：

给定一个n项多项式A(x)，最高次项n-1

给定一个m项多项式B(x)，最高次项m-1

找到一个D(x)以及R(x)使得：

A(x)=D(x)B(x)+R(x) % x^(n-m+1)

D(x)代表多项式的商，R(x)代表取模

商的项数：(n-m+1)

余数的项数：(m-1)

因此整体运算过程要对x^(n-m+1)取模，取模的做法是每次运算只计算前n-m+1项

如何计算除法？

设Ar(x)=x^n\*A(1/x)

来拿一个多项式来试试看，比如说

A(x)=x3+2x2+4x+1A(x)=x3+2x2+4x+1

那么将刚刚的操作应用上来就会变成

AR(x)=x3A(1x)=1+2x+4x2+x3

事实上就是把系数翻转而已

翻转的多项式满足如下等式，推演过程也不难略

Ar(x)= Dr(x) Br(x)

Dr(x)= Ar(x)(x)

Dr就是商的翻转,求法就是求B的翻转的逆，再和A的翻转乘

最后要把Dr翻回来成D

至于求余数R(x) 只要把D带入，就可以