不一般的DFT

绍兴市第一中学 周雨扬

摘 要

快速傅里叶变换是信息学奥赛中的一种非常有用的算法。本文介绍了快速傅里叶变换的一些比较巧妙的应用。本文介绍了bluestein 算法，并且在模数较小时将其应用到多项式多点求值和多项式多点插值上，同时也尝试利用bluestein算法解决一些比较有代表性的关于多项式的问题。

1 引言

快速傅里叶变换是一种非常有用的算法。利用快速傅里叶变换我们可以在$O(n\log n)$的时间复杂度内计算出两个多项式的乘积。同时可以利用多项式乘法解决很多种关于多项式的问题，例如多项式求逆，多项式exp等等。

但是存在一些特殊的问题，他们虽然可以使用快速傅里叶变换解决，但是需要一些巧妙的转化。本文集中对某一类特殊的问题的做法进行了归纳与总结，并且尝试将其应用到更加常见的问题上。

本文主要分为三个部分，第一部分介绍了一些关于傅里叶变换的概念。第二部分从几个例题入手，介绍了bluestein 算法的推导思路以及其较为基础的应用。第三部分围绕多项式多点求值和多项式多点插值，通过两个经典算法问题展示了bluestein 算法在模数较小时的用途，并且尝试将这种做法推广到更加一般的问题上。

2 相关定义和说明

2.1 相关约定

在本文中，我们采用符号$|f(x)|$表示多项式$f(x)$的次数。这里我们约定任意

非零次多项式的最高项系数非0。

在本文中，我们定义多项式$f(x)$在p处的点值为将p代入$f(x)$中得到的结果。

在本文中，我们采用符号ω_n表示n次单位根，也就是$\cos\left(\frac{2\pi}{n}\right)+\sin\left(\frac{2\pi}{n}\right)\mathrm{i}$。

2.2 离散傅里叶变换(DFT)

离散傅里叶变换是将$\omega_n^0,\omega_n^1,\cdots,\omega_n^{n-1}$依次代入多项式$f(x)$中得到的长度为$n$的点值序列$a$。在本文中，我们称离散傅里叶变换中的参数$n$为模长。在信息竞赛中离散傅里叶变换一般需要保证$n>|f(x)|$。

在已知长度为n点值序列a的情况下，我们构造多项式$g(x)=\sum_{i=0}^{n-1}a_ix^i$，并且求出将$\omega_n^0,\omega_n^{-1},\cdots,\omega_n^{-(n-1)}$依次代入$g(x)$中得到的长度为$n$的点值序列$b$，则存在$f(x)=\frac{1}{n}\sum_{i=0}^{n-1}b_ix^i$。证明可以见该博客[1]，此处略去。因此我们可以利用两个多项式在$\omega_n^0,\omega_n^1,\cdots,\omega_n^{n-1}$的点值来快速计算两个多项式相乘的结果。

快速傅里叶变换(FFT)是一个基于分治的离散傅里叶变换的优化。单次运行时间复杂度为$O(n\log n)$。在下文中会讲述其做法。

2.3 一般模数快速傅里叶变换(MTT)

假设我们需要求解$A(x)\times B(x)$的各项系数对$p=10^9+7$取模的结果。如果直接采用离散傅里叶变换求解，运行结果的绝对值会达到$p^2\times|A(x)|$级别，在精度上无法接受。

考虑将多项式的各项系数拆成$xC+y(0\le y<C)$的形式，其中C是一个正整数。据此我们可以将多项式$A(x)$拆成$A_1(x)C+A_2(x)$，其中$A_2(x)$各项系数均为$[0,c-1)$内的正整数。类似的对于$B(x)$也拆成$B_1(x)C+B_2(x)$。此时两个多项式乘积为$A_1(x)B_1(x)C^2+(A_1(x)B_2(x)+A_2(x)B_1(x))C+A_2(x)B_2(x)$。对于$C$次数不同的那些项，我们采用上面的方法直接求解。取$C=\sqrt{p}$时,运行多项式乘积产生的结果的绝对值只有$p\times|A(x)|$，在精度上可以接受。

或者找到3个不同的质数，使得其满足$p\times 2^k+1(k\ge 22)$的形式，在求出原根后使用快速傅里叶变换分别求解，最后使用中国剩余定理合并答案。该算法在常数上较劣。

[1]博文链接: https://www.cnblogs.com/RabbitHu/p/FFT.html

在毛啸学长2016年的集训队论文中有提到对于第一种算法的优化，有兴趣的读者可以自行查阅

3 一般模长DFT

3.1 引入

例题一[2]

题意

有一个无限循环的序列$x_1, x_2, \cdots$,循环节为n。对这个序列进行1次操作后，操作后的序列x'循环节仍为n，且满足$x'_i = x_i + x_{i+1}$。

有q次询问，每次给定n, m, k, i, p，求当序列的循环节为n，初始只有$x_i = v$，其他$x_j = 0(j \neq i)$,，操作k次以后，$x_m(m \leq n)$对p取模的值。保证n次单位根在模p意义下存在。

$k \leq 2 \times 10^9, p \leq 2 \times 10^9, \Sigma n \leq 10^6$

做法

考虑答案的组合意义后对其进行简单的单位根反演。

$$\begin{aligned}\sum_{i=0}^{\infty}\binom{K}{in+m} &= \sum_{i=0}^{K}\binom{K}{i}\frac{\sum_{j=0}^{j\leq n-1}\omega_n^{(i-m)j}}{n}\\ &= \frac{1}{n}\sum_{j=0}^{j\leq n-1}\omega_n^{-mj}\sum_{i=0}^{K}\binom{K}{i}\omega_n^{ij}\\ &= \frac{1}{n}\sum_{j=0}^{j\leq n-1}\omega_n^{-mj}(1+\omega_n^{j})^K\end{aligned}$$

求出单位根之后暴力进行快速次幂求和即可。复杂度$O(\sum n \log K)$。

例题二

题意

[2]Source:【北大集训2019】

有q次询问，每次给定最高项次数小于n 的多项式$A(x)$，询问$A(x)^K$对$(x^n - 1)$取模后，结果的每一项系数对质数p取模的结果。保证在模p意义下存在n次单位根。

$k \le 2 \times 10^9, p \le 2 \times 10^9, \Sigma n \le 10^6$

思考

直接推式子计算答案非常麻烦。采用$O(n \log n \log K)$的多项式快速次幂复杂故过高。同时由于存在对$x^n - 1$取模的缘故，无法通过多项式exp和多项式ln来解决该问题。

由于模长为n的离散傅里叶变换本质上是进行了一次长度为n的循环卷积，因此如果能够快速进行模长恰好为n的离散傅里叶变换，就可以以一个较小的常数在$O(n(\log n + \log K))$的复杂度内解决这个问题。

但是快速傅里叶变换仅能处理模长为2^k的特殊情况，而离散傅里叶变换时间复杂度过大，因此需要更加优秀且通用的解决一般模长快速傅里叶变换的方法。

3.2 快速傅里叶变换

我们先来回顾一下快速傅里叶变换的做法。假设模长$n = 2^k$，且我们要求出将$\omega_n^0, \omega_n^1, \cdots, \omega_n^{n-1}$依次代入多项式$f(x)$中得到的长度为$n$的点值序列$a$。设$f(x) = \sum_{i=0}^{n-1} f_i x^i$。

设$f_1(x) = \sum_{i=0}^{n/2-1} f_{2i} x^i, f_2(x) = \sum_{i=0}^{n/2-1} f_{2i+1} x^i$,则有$f(x) = f_1(x^2) + x f_2(x^2)$。

考虑将ω_n^k和$\omega_n^{k+n/2}$代入多项式$f(x)$的值，分别可以得到:

$$\begin{aligned} f(\omega_n^k) &= f_1(\omega_n^{2k}) + \omega_n^k f_2(\omega_n^{2k}) \\ &= f_1(\omega_{n/2}^k) + \omega_n^k f_2(\omega_{n/2}^k) \\ f(\omega_n^{k+n/2}) &= f_1(\omega_n^{2k+n}) + \omega_n^{k+n/2} f_2(\omega_n^{2k+n}) \\ &= f_1(\omega_{n/2}^k) - \omega_n^k f_2(\omega_{n/2}^k) \end{aligned}$$

不难发现运行结果仅仅和$f_1(\omega_{n/2}^k)$与$f_2(\omega_{n/2}^k)$有关。因此可以使用$O(n)$的复杂度将其转化为两个大小为$n/2$的子问题。运用主定理不难得到时间复杂度为$O(n \log n)$。这就是一般所说的快速傅里叶变换(FFT)。

3.3 基于分治的快速傅里叶变换

类比于快速傅里叶变换，我们考虑将其扩展到更加一般的模长。一般的，现在假设我们需要计算$V(j)=\sum_{i=0}^{i<n}a_i\omega_n^{ij}$的值。

类似于上面的思路，设n的最小质因子为d,设$m=n/d, j=pd+q(0\le p<m, 0\le q<d), i=xm+y(0\le x<d, 0\le y<m)$,将上述定义代入DFT式子，得到:

$$\begin{aligned}V(pd+q)&=\sum_{i=0}^{i<n}\omega_n^{i(pd+q)}a_i\\&=\sum_{i=0}^{i<n}\omega_n^{ipd}\omega_n^{iq}a_i\\&=\sum_{i=0}^{i<n}\omega_m^{ip}\omega_n^{iq}a_i\\&=\sum_{y=0}^{y<m}\sum_{x=0}^{x<d}\omega_m^{(xm+y)p}\omega_n^{(xm+y)q}a_{xm+y}\\&=\sum_{y=0}^{y<m}\sum_{x=0}^{x<d}\omega_m^{yp}\omega_d^{xq}\omega_n^{yq}a_{xm+y}\end{aligned}$$

依次枚举q的取值，则$\Sigma_{x=0}^{x<d}\omega_d^{xq}\omega_n^{yq}a_{xm+y}$是一个仅和$y$有关的常数，我们将其设为$B(y)$，并且代入原式，此时有:

$$\begin{aligned}V(pd+q)&=\sum_{y=0}^{y<m}\sum_{x=0}^{x<d}\omega_m^{yp}\omega_d^{xq}\omega_n^{yq}a_{xm+y}\\&=\sum_{y=0}^{y<m}\omega_m^{yp}B(y)\end{aligned}$$

对照最初的DFT式定义，不难发现在q确定时我们将其转化为了恰好1个大小为m的子问题，这启发我们可以通过递归的方法解决这个子问题。

考虑时间复杂度。设解决模长为n的DFT问题的时间复杂度为$F(n)$,则根据算法定义我们有$F(n)=dF(\frac{n}{d})+O(dn)$,且$F(1)=O(1)$。

设n的质因子分解形式为$\prod a_i^{p_i}$，同时根据n质因子分解形式设函数$\Omega_0(n)=\sum a_ip_i$。则对于$F(1)$满足$F(n)=O(n\Omega_0(n))$,且若对于所有$<n$的整整数均满足$F(n)=$

$O(n\Omega_0(n))$,则有$F(n)=O(\Omega_0(n/d)n)+O(dn)$,不难发现其仍然满足$F(n)=O(n\Omega_0(n))$。据此我们即可计算其复杂度。

但是$\Omega_0(n)$在最坏情况下仍然可以取到$O(n)$级别，因此我们需要一个复杂度更加优秀的算法。

3.4 基于卷积的快速傅里叶变换

让我们再次回到DFT和IDFT的式子上来考虑这个问题。

$$V(j)=\sum_{i=0}^{i<n}\omega_n^{ij}a_i$$

考虑$i\times j$的实际含义，可以看成有两堆物品，第一堆大小为i，第二堆大小为j，从每堆中选出一个物品的方案数。

在这个方案数上考虑进行容斥。用两堆物品合并，并在其中选择两个不记顺序的互异元素的方案数，减去第一堆中选两个不记顺序的互异元素的方案数和第二堆中选两个不记顺序的互异元素的方案数的和，即为从每堆中各自选出一个物品的方案数。转化成数学公式即为:$\binom{i+j}{2}-\binom{i}{2}-\binom{j}{2}=i\times j$。将上面的容斥式子代入DFT式，可以得到:

$$V(j)\omega_n^{\binom{j}{2}}=\sum_{i=0}^{i<n}a_i\omega_n^{-\binom{i}{2}}\omega_n^{\binom{i+j}{2}}$$

这个式子可以看成是$C(x)=\sum\omega_n^{-\binom{i}{2}}x^i$对$B(x)=\sum a_i\omega_n^{\binom{i}{2}}x^i$做一次减法卷积的结果，然后对每一项系数乘上一个对应常数的值。此时我们并不关心卷积的模长带来的影响，只关心卷积的结果，而不关心卷积的模长问题，因此可以直接采用MTT求解。时间复杂度$O(n\log n)$。

类似的，对于IDFT式也代入上面的容斥式子，可以得到:

$$V(j)\omega_n^{-\binom{j}{2}}=\sum_{i=0}^{i<n}a_i\omega_n^{\binom{i}{2}}\omega_n^{-\binom{i+j}{2}}$$

类似的，这个式子仍可以被看成是一次减法卷积，然后对每一项系数乘上一个对应常数的值，因此仍然可以直接一次MTT求解。时间复杂度$O(n\log n)$。这个算法也被称为bluestein 算法或者Z变换。

事实上，对于任意的非零数字a，我们都可以通过这种思路来计算$V(j)=\Sigma_{i=0}^{i<n}x_j a^{ij}$的值。我们只需要保证存在$a^{ij}$以及其逆元即可。也就是说，对于复数该算法仍然适用。

- 小优化技巧：原本减法卷积部分的模长需要开到$3*n$级别，但是根据FFT加速多项式乘法本质是进行了一次循环卷积的特性，由于只关心中间n个元素的正确性，因此将最后n个元素和前面的n个元素在结果中重合对答案的正确性不会产生任何的影响。因此只需要将模长开到$2*n$级别即可。

3.5 例题

例题三[3]

题意

定义两个简单无向图$G_1=(V_1,E_1),G_2=(V_2,E_2)$的乘积为一个新的图$G_1\times G_2=(V^*,E^*)$。其中新的点集$V^*$满足$V^*=\{(a,b)|a\in V_1,b\in V_2\}$,其中新的边集$E^*$满足$E^*=\{((u_1,v_1),(u_2,v_2))\mid (u_1,u_2)\in E_1,(v_1,v_2)\in E_2\}$。

给定正整数n，以及n个正整数$m_1,m_2,\cdots,m_n$。你需要求出新图$H=(((G_1\times G_2)\times G_3)\times\cdots)\times G_n$的期望连通块数量对998244353取模的结果。其中G_i所有包含m_i个节点的图中等概率随机生成。

$n,m_i\le 100000$

做法

我们考虑给我们一组G_k的序列之后如何计算连通块数量。

首先考虑在每个图中选一个点，如果某个选的点数的度数为0（我们称其为“孤立点”），则这个点序列在H上对应的点是孤立点。否则我们考虑每个图中选择一个大小> 1 的连通块。考虑这些连通块的乘积得到的连通块有多少。注意到现在每个点都有邻边，且是无向图，因此我们可以在一条边上反复走。两个点序列之间的可达性可以简化为路径长度的奇偶性。

如果两个点在一个存在奇环的图中，那么显然奇数长度和偶数长度的路径都有。如果两个点在一个二分图中，那么这和他们是否在同一部中有关。因此我们可以得到：如果选的这n个连通块中有k个不存在奇环，那么这些连通块的乘积将会给答案贡献$2^{\max(k-1,0)}$个连通块。因此我们只需要知道全体大小为m_k的

[3]Source:【UOJ498】新年的追逐

图可以有多少个孤立点，多少个无奇环的连通块，多少个连通块，则可以由此算出答案。

记无向简单图的EGF 为$G(x)=\sum_{n\geq 0}\frac{2^{\binom{n}{2}}x^n}{n!}$，则联通图的EGF 为$C=\ln G$。不难得到连通块数量的EGF 由枚举连通块如何插入一个图得到，即为$G\ln G$。

我们考虑染色二分图的EGF$B=\sum_{n\geq 0}\sum_{m\geq 0}\frac{\binom{n+m}{n}2^{nm}x^{n+m}}{(n+m)!}$，则无奇环的连通块显然恰有2 种方法染色，可以得到EGF 为$\frac{\ln B}{2}$，无奇环连通块数量可以通过$\frac{G\ln B}{2}$ 表示。这里我们可以采用`bluestein` 算法来优化求染色二分图的EGF 的过程。

时间复杂度$O(n\log n)$。

例题四[4]

题意

给定长度为n的非负整数序列b_i和c_i，下标均从0开始。

已知非负整数序列a满足$c_i=\sum_{k=0}^{n-1}(b_{k-i \bmod n}-a_k)^2$,求$a_i$的所有合法解。保证$b$的所有长度为$n$的循环移位线性无关。

$n\leq 10^5, b_i\leq 10^3, c_i\leq 5\times 10^6$。

做法

为了方便描述，接下来所有的序列都是循环节为n的无线序列。

由于$(a-b)^2=a^2+b^2-2ab$，则$c_k-c_{k-1}=\sum_{i=0}^{n-1}b_i(a_{i+k}-a_{i-k+1})$。设$c'_k=c_k-c_{k-1}, a'_k=a_k-a_k-1$，则有$c'_k=\sum_{y-x \bmod n=k}b_x a'_y$。因此$c'_k$可以被看成是$\sum b_i x^i$对$\sum a'_i x^{n-i}$做模长为$n$的卷积的结果。利用模意义下的bluestein 算法我们可以快速计算出唯一一组合法的a'_i。

设$a_0=x$，则其余数字可以使用x表示，此时即可列出一个一元二次方程，就可以解出不超过两个合法解，直接代入检验合法性即可。

时间复杂度$O(n\log n)$。

4 DFT在多点求值上的应用

4.1 一般思路

多项式多点求值问题是多次询问多项式$f(x)$在值y处的点值的问题。

[4]Source:CF 901E Cyclic Cipher

由于离散傅里叶变换是将$\omega_n^0, \omega_n^1, \cdots, \omega_n^{n-1}$依次代入多项式$f(x)$中得到的长度为$n$的点值序列$a$。因此在解决多项式多点求值问题时，如果询问的点值满足某些特殊性质，我们就可以将其重排列之后，采用bluestein 算法加速多项式多点求值的过程。

例题五[5]

题意

给定n次多项式$f(x) = \sum_{i=0}^{i\le n} f_i x^i$。$Q$次询问，第$i$次询问$f(q_i)$对$p = 998244353$取模后的结果。$q_i$按照如下方式生成:

$\forall 1 \le i \le Q, q_i = (q_{i-1} \times a + b)\ mod\ 998244353$。

$1 \le n \le 2.5 \times 10^5, 1 \le Q \le 10^6, 2 \le a < 998244353, 0 \le q_0, b < 998244353$。

做法

假设我们已知任意n次多项式$f(x)$,则我们可以:

- 在$O(n)$的复杂度内找到唯一一个次数不超过n次多项式$g(x)$ 满足$g(x) = f(x * k)$
- 在$O(n \log n)$的复杂度内找到唯一一个次数不超过n 次多项式$h(x)$ 满足$h(x) = f(x + k)$。

观察到询问的值比较的特殊，我们尝试利用一些上述变换，通过加减乘除操作将询问的值转化成比较优美的形式。

首先归纳得到$q_i = q_0 a^i + \Sigma_{j=0}^{j<i} b a^j$。接下来进行如下变换:

(1) 将右式乘以$a - 1$，得到$q_i = q_0(a-1)a^i + ba^i - b$。

(2) 将右式加上b，得到$q_i = q_0(a-1)a^i + ba^i$。

(3) 将其除以$q_0(a-1) + b$，得到$q_i = a^i$

注意在$q_0(a-1) + b$为0时，此时满足$q_i = q_0$，因此只需要求出q_0处的单个点值即可。

[5]Source:【UOJ500】任意基DFT

否则在给定数据范围下，上述所有运算均合法，因此我们可以对$f(x)$做类似的变换，即求出n次多项式$g(x)$满足$g(x) = f(\frac{x(q_0(qx-1)+qy)-qy}{qx-1})$。根据上述变换的性质，有且仅有一个满足条件的次数不超过n次的多项式$g(x)$。

现在我们需要求出多项式$g(x)$在$a^1, a^2, \cdots, a^Q$处的点值。即需要求出$V(i) = \Sigma_{j=0}^{j\le n} g_j a^{ij}$。不难发现求解的式子满足bluestein 算法的各项性质，因此可以使用该算法求解。

时间复杂度$O((n+Q)\log(n+Q))$。

例题六[6]

题意

给定n次多项式$f(x) = \sum_{i=0}^{i\le n} a_i x^i$。$Q$次询问,每一次给定正整数$y$,询问$f(y)$对$p = 786433(3 * 2^{18} + 1)$取模的值。

$n, Q \le 250000, 0 \le y < p$。

做法

考虑离散傅里叶变换的实际含义，不难发现模长为n的DFT可以看成是将$\omega_n^0, \omega_n^1, \cdots, \omega_n^{n-1}$ 代入多项式$A(x)$产生的点值。

由于p是质数，因此p存在原根，这说明存在g满足$\omega_{p-1} \equiv g(\mathrm{mod}\ p)$。同时根据原根的定义，$g^0, g^1, \cdots, g^{p-2}$ 在模意义下互不相同且恰能取到所有与p互质的小于p的正整数。也就是形成了一个$[1, p-1]$ 的排列。

因此如果我们对$A(x)$跑一轮模长为$p-1$的DFT，并且将得到的点值重排列之后就可以得到$1, 2, \cdots, p-1$的点值。同时，0处的点值显然为a_0。因此所有点值都已经被求出。

时间复杂度$O(p\log p)$。

4.2 推广

我们尝试将例题六的模数拓展到更加一般的情况。现在假设我们需要求出点值对$p = q^c (c \ge 1)$取模的结果,其中q为大于2的任意质数。

由于q^c存在原根，这说明存在g满足$\omega_{\phi(q^c)} \equiv g(\mathrm{mod}\ q^c)$。同时根据原根的定义，$g^0, g^1, \cdots, g^{\phi(q^c)-1}$ 在模意义下互不相同且恰能取到所有与q^c互质的小于q^c的

[6]Source:CODECHEF POLYEVAL

正整数。这些点的点值我们仍然可以使用一次DFT计算得出。

对于与q^c不互质的整数，设其为x，则显然有$x^c \equiv 0 (\mathrm{mod}\ q^c)$,这说明我们仅需要考虑小于$c$次的项产生的贡献即可，因此直接暴力计算前$c-1$项的值就可以算出点值。由于$c = O(\log p)$，该部分时间复杂度仍为$O(p \log p)$。

特殊的，对于$p = 2^c (c \geq 3)$的特殊模数，虽然其不存在原根，但是仍然可以证明$\pm 3^0, \pm 3^1, \cdots, \pm 3^{2^{c-2}}$在模意义下互不相同且均与2互质，证明过程较为复杂此处略去。这些点的点值我们仍然可以使用两次DFT计算得出。类似的，与2不互质的仍然只需要考虑小于c次的项。

但是由于快速傅里叶变换的模长与2不互质，在IDFT时候会出现问题。因此我们需要通过将快速傅里叶变换部分的模数乘以模长，同时将最后的结果暴力除以模长解决。或者通过将FFT部分的模长设置为3^k，然后利用基于分治的快速傅里叶变换解决。因此仍然可以在$O(p \log p)$的时间复杂度内解决该问题。

若使用中国剩余定理合并不同q^c的答案，则对于任意模数均可做到$O(p \log p)$的多项式多点求值。

5 DFT在多点插值上的应用

多项式多点插值问题是给定多项式$f(x)$在某些位置y上的点值，询问$f(x)$的各项系数的问题。

由于DFT可以在多项式多点求值上得到良好的应用，因此我们考虑将其拓展到其逆操作，即多项式多点插值上。

5.1 连续点值的多点插值

题意

给定n次多项式$f(x)$在$0 \sim n$处的点值序列a，询问$f(x)$每一项系数对p取模的结果。

$0 < n < p \leq 500000$,p为质数。

算法

因为多点求值的逆操作为多点插值。DFT的逆操作为IDFT。如果多点求值看成是一次DFT操作，那么这提示我们可以尝试使用一次IDFT来解决多项式多点插值问题。

具体的，首先我们根据$0 \sim n$的点值计算出$0 \sim p-1$的点值。这里可以通过点值与多项式的下降幂性质之间的互相转换实现，这一部分不再展开。

由于我们已知$1 \sim p-1$的点值，我们可以通过一次IDFT确定出一个不超过$p-1$次多项式$f(x)$,使得$f(x)$在$1 \sim p-1$处的点值恰好为$a_1, a_2, \cdots, a_{p-1}$。这里部分正确性可以由DFT与IDFT的正确性得到。因此现在只需要构造多项式$h(x)$满足$h(x) \equiv [x=0] (\mathrm{mod} p)$。不难发现此时$f(x)+h(x)(a_0-f(0))$即为答案。

我们可以构造以下多项式$h(x)=\sum_{j=0}^{j<p-1} x^j$。在$x=0$时其值为1,在$x=1$是其值为$p$，在模意义下值为0。在$x$大于1时根据等比数列求和公式，答案为$\frac{x^{p-1}-1}{x-1}$。根据费马小定理$x^{p-1} \equiv 1 (\mathrm{mod}\ p)$,因此在模意义下值仍为0。直接将其代入上面的式子中即可得到答案。

时间复杂度$O(p \log p)$。

5.2 非连续点值的多点插值

题意

给定n次多项式$f(x)$在$a_1, a_2, \cdots, a_{n+1}$处的点值$g_1, g_2, \cdots, g_{n+1}$，询问$f(x)$每一项系数对$p$取模的结果。

$0 < n < p \leq 300000$,p为质数,$n < p-2$。保证$a_i > 0$，且a_i互不相同。

算法

根据拉格朗日插值，我们可以很方便的计算出答案为$\sum_i (g_i \sum_{j \neq i} \frac{x-a_j}{a_i-a_j})$。

设$F(x)=\Pi_i(x-a_i)$,并对式子稍作转化，答案可以化简为$(\sum_i \frac{g_i}{(x-a_i)F'(a_i)})F(x)$。

对于求解$F'(a_i)$的那部分，我们可以通过多项式多点求值来解决，最后乘以$F(x)$的部分可以放到最后处理。因此现在我们将其转化为两个比较有价值的子问题:

- 求解$F(x)=\Pi_i(x-a_i)$。
- 求解$G(x)=\sum_i \frac{v_i}{x-a_i}$。

子问题1

$$F(x)=\prod_i(x-a_i)$$
$$\frac{F(x)}{\prod_i -a_i}=\prod_i(-\frac{1}{a_i}x+1)$$
$$\ln\left(\frac{F(x)}{\prod_i -a_i}\right)=\sum_i \ln(-\frac{1}{a_i}x+1)$$
$$\ln'\left(\frac{F(x)}{\prod_i -a_i}\right)=\sum_i\sum_j x^j\frac{1}{(a_i)^{j+1}}$$

根据上式子问题1可以通过一次多项式exp和一次积分，在$O(p\log p)$的复杂度内转化为子问题2。接下来我们仅讨论子问题2的做法。

子问题2

$$\begin{aligned}G(x)&=\sum_i\frac{v_i}{x-a_i}\\&=\sum_i v_i\sum_j x^j\left(\frac{1}{a_i}\right)^{j+1}\\&=\sum_j x^j\sum_i v_i\left(\frac{1}{a_i}\right)^{j+1}\\&=\sum_j x^j\sum_k\sum_i[\omega_{p-1}^k=a_i]v_i\omega_{p-1}^{-k(j+1)}\end{aligned}$$

设$H(k)=\sum_i[\omega_{p-1}^k=a_i]v_i$, 则有

$$G(x)=\sum_j x^j\sum_k H(k)\omega_{p-1}^{-k(j+1)}$$
$$G(x)x=\sum_j x^j\sum_k H(k)\omega_{p-1}^{-kj}$$

观察最后式子，不难发现其满足bluestein 算法的性质。因此仍然可以使用bluestain 算法加速计算$G(x)x$ 的值。

因此两个子问题我们都可以在$O(p\log p)$的时间复杂度内解决，总复杂度仍为$O(p\log p)$。

特殊情况

对于$a_i = 0$的情况，由于在解决子问题1,2时，我们的推导过程使用到了$\frac{v_i}{x}$以及其导数,因此上面的推导会产生严重的错误。

其中一种解决方案是利用之前在例题三提到的构造$g(x) = f(x + k)$的方式，将a_i整体平移之后使得对于所有a_i其值全部非0,据此我们可以计算出唯一的g_i。最后将计算出来的$g(x)$重新平移回来即可。

另外一种解决方案是在插值时忽略$a_i = 0$处的点值进行插值，最后加上若干倍的$\Pi_{a_i \neq 0} x - a_i$使得$f(0)$处的点值正确。这种方法也被称为部分拉格朗日插值法。

5.3 拓展

子问题2可以看成是给定列向量之后将其右乘一个行列式非0的Vandermonde矩阵的结果。事实上我们也可以利用上述算法在模数较小时解Vandermonde矩阵。

设我们需要解方程组:$G(j) = \sum_i a_i x_i^j$,类似于问题2,设$H(k) = \sum_i [\omega_{p-1}^k = x_i] v_i$,则方程组可以转化为:$G(j) = \sum_k H(k) \omega_{p-1}^{kj}$。

类比于之前探讨过的问题，不难发现，我们将问题转化到了连续点值的多项式多点插值上，因此可以直接利用该算法，可以在$O(p \log p)$的时间复杂度内解决该问题。

6 后记

上述大部分算法都是在模域下的bluestein 算法的应用。事实上在2003年，学界提出了一种时间复杂度为$O(n \log n)$ 的bluestein 算法的逆运算。有兴趣的读者可以自行查阅相关论文。

7 展望

本文提到了一些解决快速傅里叶变换问题的方法和技巧，但在浩瀚的算法海洋中仍只是冰山一角。希望未来能有更多的bluestein 算法的应用被发现和发明，也希望有更多有意思的快速傅里叶变换题出现在信息学奥赛中。

8 致谢

感谢中国计算机学会提供学习和交流的平台。
感谢绍兴一中的陈合力老师和董烨华老师的关心和指导。
感谢国家集训队教练高闻远的指导和帮助。
感谢戴江齐同学对于非连续点值多点插值的算法方面做出的启发。
感谢邓明扬同学，孔朝哲学长为本文验稿。
感谢其他对我有过帮助和启发的老师和同学。
感谢父母对我的关心、支持与无微不至的照顾。

9 参考文献

[1] 张家琳, 《多项式乘法》, 2002年集训队论文.

[2] 毛啸, 《再探快速傅里叶变换》, 2016年集训队论文.

[3] Bluestein, L. (1970). ”A linear filtering approach to the computation of discrete Fourier transform”.