多项式（水）题选讲

镇海中学 罗煜翔

前置技能

- FFT，NTT。
- 多项式求逆、除法。
- 常见多项式算法的 $O(n^2)$ 实现。
- 基础的数论和组合计数知识。

例 1. 欧拉数

- 给定 n，对所有 $0 \le k \le n$，求欧拉数 $\left\langle {n \atop k} \right\rangle$，模 998244353。
- $n \le 10^7$。5s

- 由于欧拉数的对称性，只需要求前一半。
- 利用 $\left\langle {n \atop k} \right\rangle = \sum_{i=0}^{k} \binom{n+1}{i}(k+1-i)^n(-1)^i$，只需要做一次多项式乘法。但需要长度 n，超过了 2^{23}。
- 可以利用分治乘的方法变成一半长度，也可以用分治FFT做。

半在线卷积

- 在很多问题中，我们都要处理一下的半在线卷积：
  - $f = g \times h$，其中 g 给定且常数项为 0，在计算出 f_i 后给出 h_i。
- 可以使用分治算法解决这个问题：
  - 设 solve(l,r) 表示只考虑 f, h 的下标在 $[l, r]$ 范围内的情况。
  - 取区间中点 $mid = \left\lfloor \frac{l+r}{2} \right\rfloor$，先递归处理 $[l, mid]$，利用 $[l, mid]$ 的 h 计算对 $[mid+1, r]$ 的和，然后递归处理 $[mid+1, r]$。取最大的 2 的幂代替中点可以优化常数。
- 事实上，这个分治可以改成每次分 K 叉，则每次需要计算 $O(K^2)$ 对转移。但注意到多项式只有 $O(K)$ 种，且次数为 $O\left(\frac{n}{K}\right)$，于是复杂度为 $O(n \log n + nK)$，取 $K = O(\log n)$ 则每层复杂度不变，但每次能分 K 叉，总的复杂度变成了 $O\left(\frac{n \log^2 n}{\log \log n}\right)$。对于部分 n 甚至比 FFT 还要快！

例 2 、因式分解

- 给定一个保证能分解成形如 $\prod_{i=1}^{m}(x+d_i)^{l_i}$ 且 l_i, d_i 均互不相同的 n 次多项式，将它因式分解。上述均模 $P=998244353$。
- $n \le 2000$。3s

- 由于在模意义下，$x^P - x = \prod_{i=0}^{P-1}(x-i)$，所以将原多项式和 $x^P - x$ 求 gcd 就可以保证 $l_i = 1$。这可以通过先计算 $x^P - x$ 模原多项式再做 gcd 完成。
- 在保证 $l_i = 1$ 后，设当前多项式为 f，每次随机一个 n 次多项式 g，则 $\gcd\left(f, g^{\frac{P-1}{2}}+1\right)$ 有很大概率与 f 有非平凡因子，然后递归处理即可。

例 2 、因式分解

- 最后用得到的 d_i 试除即可得到所有的 l_i。
- 如果均采用朴素的多项式操作，第一部分时间复杂度为 $O(n\log P)$，第二部分由于次数最多 $O(\sqrt{n})$，于是复杂度为 $O(n\log n\log P)$，最后一部分时间为 $O(n^2)$。
- 将求 $x^P - x$ 模原始多项式的步骤用 FFT 优化，就可以做到 $O(n^2)$。

例题3、$\mathbb{F}_{2^{32}}$ 上的多项式乘法

- 给定两个 $\mathbb{F}_{2^{32}}$ 上的 n 次多项式，求他们的乘积。
- $n \le 10^5$。3s

- 显然可以利用三维 FFT 解决，但常数较大。
- 考虑进行分治，问题在于如何快速计算两个元素的乘法。
- 一种做法是利用 Nim 积，预处理 65535 内的原根和指数对数表，这样可以 $O(1)$ 计算乘法。而对于所有数，可以用 3 次 65535 内的乘法解决。常数和长度都比较小。
- 还可以利用 $O(n\log n\log\log n)$ 的多项式乘法算法。

例题 4 、求和

- 给定互质的正整数 p,q，求所有无法表示成 $px+qy\ (x,y\ge 0)$ 的数的 k 次方和。模 998244353。多组数据。
- $p,q\le 10^{18}, k\le 10^5, \sum k\le 10^6$。1s

- 定义：$f(n)=[n\ge 0\wedge \forall i,j\in\mathbb{N}, ai+bj\ne n]$，那么要求 $\sum_n f(n)n^k$。
- 由伯努利数的经验，我们转而求形式幂级数 $\sum_n f(n)e^{nx}$ 的 k 次项系数。
- 事实上一个布尔值是不太方便的，计数更接近我们平常的操作。
- 考虑 n 表示为 $ai+bj$ 的方案数 $g(n)$，当 $0\le n<ab$ 时 $g(n)=1-f(n)$。
- 容易发现 $g(n)$ 满足 $g(n+a+b)=g(n+a)+g(n+b)-g(n)$。
- 于是对 $0\le n<ab-a-b$，$f(n+a+b)=f(n+a)+f(n+b)-f(n)$。
- 当 $n=ab-a-b$ 时显然 $f(n+a+b)=f(n+a)+f(n+b)-f(n)+1$。

例题 4 、求和

- 这表明，数列 有递推式：
- 当 的时候，会有其他的特判。
- 。
- 若生成函数为 ，则：
- ，其中 的次数小于 。 表示的就是其他的特判。
- 现在我们只需要求 了。

例题 4 、求和

- 事实上 。
- 考虑 ，就是当 时 的值，显然：
- 。
- 。
- 从而 。
- 由此可得：
- 事实上，右边已经是 次的多项式了，所以两端是相等的。

例题 4 、求和

- 现在我们就得到了 $R(x)$，由此可得 $F(x)$：
- $F(x)=\frac{x^{ab}-1}{(x^a-1)(x^b-1)}-\frac{1}{x-1}$。
- 我们要求的是 $\sum_n f(n)e^{nx}=F(e^x)$，所以答案就是：
- $F(e^x)=\frac{e^{abx}-1}{(e^{ax}-1)(e^{bx}-1)}-\frac{1}{e^x-1}$。
- 我们只需要多项式乘法和多项式求逆就可以解决这个问题。
- 具体实现时需要现在两端乘 x 保证可以分开计算，多组数据可以预处理伯努利数使得每次不需要求逆。

- 现在我们就得到了 ，由此可得 ：
- 。
- 我们要求的是 ，所以答案就是：
- 。
- 我们只需要多项式乘法和多项式求逆就可以解决这个问题。
- 具体实现时需要现在两端乘 保证可以分开计算，多组数据可以预处理伯努利数使得每次不需要求逆。

Thanks