线性筛与积性函数

清华大学 何昊天
kiana810@126.com

质数

- 因数只有1和自己的数称为质数
- 几个比较常用的结论：
  - 质数的个数是无限的，可用$\pi(n)=n/\ln(n)$来估计质数的个数
  - 算术基本定理：任何一个大于1的正整数，都可以唯一分解为有限个质数的乘积形式
  - 若n为质数，则$\forall 1\le i<n$，$\gcd(i,n)=1$
  - 费马小定理：若p是质数，$\gcd(a,p)=1$，则$a^{p-1}\equiv 1 \pmod p$（此定理可进一步推广为欧拉定理，其逆命题不成立）
  - 质数分布确实有一些规律，但质数分布的规律从来都不是信息学竞赛中解决相关问题的突破口

质数判定

- 最朴素的方法：枚举2~√n，依次判断其是否为n的因数，时间复杂度O(√n)
- 费马判别法：选取k个不同的变量a($2 \le a < n$)，依次计算$a^{n-1} \bmod n$是否等于1，若不是，则证明n是合数，若选择的a均满足这个条件，则n只有极小的概率不是质数，时间复杂度O(klogn)，k往往取20左右即可
- 原理：利用了费马小定理的逆命题，虽然其逆命题是错误的，但是对于合数有一定概率是不满足的，多次判断即可增加正确率，所以这是一种随机算法
- 费马判别法在某些特定的数（如卡迈克尔数）上正确率将大幅降低，但可以进一步结合质数其它必要但不充分的性质来设计类似的随机算法以提高正确率，如Miller-Robin判别法等，此处不再赘述

质数的筛法

- 问题：如何找出1~n中所有的质数？
- 枚举所有数并依次使用质数判别法，可以得到一个O(n√n)或O(nklogn)的算法，但前者复杂度太高，后者是随机算法，我们对这两个算法都不太满意

- 实际上，竞赛中常用的筛法有两种：Eratosthenes筛法和Euler筛法，它们的时间复杂度都是线性或非常接近线性的，故我们称之为线性筛，它们不仅能很快的筛出1~n中所有的质数，也对后面积性函数的预处理起着重要作用
- 你可能听说过杜教筛和洲阁筛两种黑科技，它们能够在低于线性的时间复杂度内得到一些积性函数的前缀和，但这两种方法不是讲解的重点，感兴趣的同学可以自行学习，本课件之后不再赘述

Eratosthenes筛法代码

```
for (int i=2;i<=n;i++)
if (!check[i])
{
    prime[++top]=i;
    for (int j=i*2;j<=n;j+=i)
        check[j]=1;
}
```

Eratosthenes筛法

- 关键点：用每个数筛去它的所有倍数，从未被筛过的数就是质数
- 时间复杂度：对于一个数c，若从c*2开始筛数，则由简单的调和级数求和得复杂度不超过O(nlogn)，实际上，利用质数的倒数之和，可以证明该筛法的时间复杂度为O(nloglogn)，由于后者的证明较难，我们不在这里讲解
- 两个小优化：
  - ①c的枚举只用到√n，大于√n的数要么是质数，要么有一个小于√n的因数
  - ②筛数可以从c*c开始而不从c*2开始，小于c*c的倍数已经被另一个因数筛去了

Euler筛法代码

```
for (int i=2;i<=n;i++)
{
    if (!check[i])prime[++top]=i;
    for (int j=1;j<=top&&i*prime[j]<=n;j++)
    {
        check[i*prime[j]]=1;
        if (i%prime[j]==0)break;
    }
}
```

Euler筛法

- 关键点：用每个数筛去它的质数倍数，从未被筛过的数就是质数
- 时间复杂度：严格的O(n)，我们接下来证明每个数最多会被筛去一次，也就证明了这个复杂度
- 先证明每个合数一定会被筛去：
  - 由算术基本定理，$\forall n=\prod p^k$，不妨设p_0是n的最小质因数且，则一定有$n/p_0 > p_0$，且n/p_0的质因数均大于等于p_0，所以当枚举到n/p_0时，一定会将$n/p_0*p_0=n$筛去
- 再证明每个合数仅会被其最小的质因数筛去：
  - 先作与上述相同的假设，再设p_1为n的另一质因数，则必有$p_0|(n/p_1)$，故枚举到n/p_1时，我们不会根据$n/p_1*p_1=n$而将n筛去，而是会在这之前break掉，故n只会因为p_0而被筛掉
- 综上所述，Euler筛法的复杂度和正确性得证

数集与数论函数

- 我们不严格定义集合，但需要知道一些常用的数集：
  - N：非负整数，Z：整数，Z^+：正整数
  - Q：有理数，R：实数，R\Q：无理数
  - C：复数
- 函数：
  - 从一个数集A到另一个数集B的映射，即给定$\forall x\in A$，都$\exists y\in B$，使得x和y对应起来
- 数论函数：
  - 从N→C的映射，不过竞赛中我们研究的范围有限，往往只研究从N→N的映射

函数的性质

- 基本性质：单调性、有界性、凹凸性等知道即可，用处不大
- Dirichlet特征：
  - 对于一个函数$\chi(n)$，若满足：
    - $\exists k$，s.t.$\forall n\in N$，$\chi(n)=\chi(n+k)$
    - $\forall m,n\in N$，$\chi(mn)=\chi(m)\chi(n)$
    - $\chi(1)=1$
  - 则称$\chi(n)$拥有Dirichlet特征

积性函数

- 性质①：
  - $f(1)=1$
- 性质②：
  - 设$n=\prod p_i^{k_i}$，则$f(n)=f(\prod p_i^{k_i})=\prod f(p_i^{k_i})$
- 性质③：
  - 设$n=\prod p_i^{k_i}$，则$f(n)=f(\prod p_i^{k_i})=\prod f(p_i)^{k_i}$
- 满足性质①②的称为积性函数，满足性质①②③的称为完全积性函数

常用的积性函数

- 欧拉函数(φ)：$\varphi(n)$表示1~n中与n互质的数的个数
- 莫比乌斯函数(μ)：$\mu(1)=1$，若$n(n>1)$含有多重质因数，则$\mu(n)=0$，否则$\mu(n)=(-1)^r$，r表示n的质因数个数
- 除数函数(σ)：一类函数，每个非负整数x都对应一个除数函数σ_x，$\sigma_x(n)$表示n的因数的x次方之和
- 这三类函数十分重要，接下来我们将研究它们的一些性质，以及如何线性筛出这些函数的值，另外还有一些比较平凡的积性函数，我们只是借助它们作为方便的表示，故不再单独研究

欧拉函数的积性1

- 我们先来根据定义计算一些特殊的欧拉函数的取值
- 若p为质数，则$\varphi(p)=p-1$，$\varphi(p^k)=p^k-p^{k-1}$
- 我们接下来证明若$\gcd(n,m)=1$，则$\varphi(nm)=\varphi(n)\varphi(m)$
- 首先列出一张数表：

1	$m+1$	$2m+1$	...	$(n-1)m+1$
2	$m+2$	$2m+2$	...	$(n-1)m+2$
...	...	...	...	...
r	$m+r$	$2m+r$	...	$(n-1)m+r$
...	...	...	...	...
m	$2m$	$3m$	...	nm

欧拉函数的积性2

- 我们的目标是找出数表中有多少个数与nm互质
- 考虑第r行，若gcd(m,r)≠1，则整行数都与nm不互质，所以我们需要考虑的行必须满足gcd(m,r)=1，共有φ(m)行
- 考虑第r行的第k个数(k-1)m+r，显然每一行的数恰好构成了一个模n剩余系，不妨记(k-1)m+r mod n=d
- 当且仅当gcd(n,d)=1时，gcd(n,(k-1)m+r)=1，故每一行也恰好有φ(n)的数与n互质，共有φ(n)个数
- 综上所述，数表中共有φ(n)φ(m)个数与nm互质，所以φ(nm)=φ(n)φ(m)

欧拉函数的积性3

- 设$n=\prod p_i^{k_i}$，则$\varphi(n)=\varphi(\prod p_i^{k_i})=\prod \varphi(p_i^{k_i})$

  又由$\varphi(p_i^{k_i})=p_i^{k_i}-p_i^{k_i-1}=p_i^{k_i}(1-\frac{1}{p_i})$

  可得$\varphi(n)=n\prod(1-\frac{1}{p_i})$

- 至此，我们完成了对欧拉函数积性的证明，并总结出了一个非常好用的计算公式，接下来我们来研究如何线性筛预处理出1~n的欧拉函数的值

欧拉函数的线性筛

- 注意到$\varphi(p^k)=p^{k-1}(p-1)$，对于一个数n，假设p是它的最小质因数，且最大重数为k，则我们应该依次算出n/p^k，n/p^{k-1}，……，n的欧拉函数值来
- 实际上，我们会用n/p^t的p倍来筛掉n/p^{t-1}，依次类推，所以只需要在每次筛去的时候，用$\varphi(n/p^t)$乘上p即得$\varphi(n/p^{t-1})$
- 但有一个例外，当t=k时，我们不应乘以p，而应该乘以p-1，不过这并不难办，只需要在执行上述流程时，判断$n/p^t \bmod p$是否等于0，若不是，则乘以p-1，否则就乘以p
- 递归地考虑n除了p以外的其它质因数，也即n/p^k的所有质因数，它们共同贡献了$\varphi(n/p^k)$，所以这个方法是正确的

欧拉函数线性筛代码

```
for (int i=2;i<=n;i++)
{
    if (!check[i])
    {
        prime[++top]=i;
        phi[i]=i-1;
    }
    for (int j=1;j<=top&&i*prime[j]<=n;j++)
    {
        check[i*prime[j]]=1;
        if (i%prime[j]==0)
        {
            phi[i*prime[j]]=phi[i]*prime[j];
            break;
        }
        else phi[i*prime[j]]=phi[i]*(prime[j]-1);
    }
}
```

欧拉函数的推论

- 推论：当n>1时，1~n中与n互质的数的和为$n*\varphi(n)/2$
- 证明：
  - 若n≠2，则显然成立
  - 若n≠2，设gcd(n,p)=1，由更相减损术得gcd(n-p,p)=1，且一定有p≠n-p，故1~n中与n互质的数可以由此两两配对，而每一对的和都为n，所以总和为$n*\varphi(n)/2$
- 这个推论很常用，之后还会用反演推出这个推论的更完美的形式

欧拉定理

- 定理：若$n,a\in N$满足$\gcd(n,a)=1$，则$a^{\varphi(n)}\equiv 1 \pmod n$
- 证明：
  - 将1~n中与n互质的数依次排列为$x_1,x_2,\ldots,x_{\varphi(n)}$，并令$m_i=a*x_i$
  - 显然$m_1,m_2,\ldots,m_{\varphi(n)}$两两模n不同余，故我们有$m_1m_2\ldots m_{\varphi(n)}\equiv x_1x_2\ldots x_{\varphi(n)} \pmod n$
  - 将上式化简后得$(a^{\varphi(n)}-1)x_1x_2\ldots x_{\varphi(n)}\equiv 0 \pmod n$，但$x_1,x_2,\ldots,x_{\varphi(n)}$均与n互质，故必有$a^{\varphi(n)}\equiv 1 \pmod n$
- 当n为质数时，$\varphi(n)=n-1$，故费马小定理是欧拉定理的特殊形式

扩展欧拉定理

- 扩展欧拉定理在这个版块不太用得上，所以只是给出定理本身的叙述，供大家做了解与参考
- 设a,b,n∈N，若：
  - gcd(n,a)=1，则$a^b \equiv a^{b \bmod \varphi(n)} \pmod n$
  - gcd(n,a)≠1，b≤φ(n)，则用快速幂计算
  - gcd(n,a)≠1，b>φ(n)，则$a^b \equiv a^{(b \bmod \varphi(n))+\varphi(n)} \pmod n$

莫比乌斯函数的积性

- 首先回忆一下莫比乌斯函数的定义：
  - 若n=1，则μ(n)=1
  - 若n没有多重质因数，则μ(n)=$(-1)^r$，其中r表示n的质因数个数
  - 其它情况，μ(n)=0
- 只要分别验证几种情况，易证莫比乌斯函数是积性函数但不是完全积性函数，至于莫比乌斯函数究竟有什么含义，我们将在之后研究反演的时候慢慢体会，接下来就来研究莫比乌斯函数如何线性筛预处理

莫比乌斯函数的线性筛

- 类似于欧拉函数的线性筛，我们只需要简单区分一个质数是否是当前枚举的数的因数即可
- 对于一个数n，假设p是它的最小质因数，且最大重数为k，假设我们正准备用n/p^t的p倍来筛掉n/p^{t-1}，若$t\neq k$，则p为n的多重质因数，则$\mu(n/p^{t-1})=0$，否则$\mu(n/p^{t-1})=\mu(n/p^t)*(-1)$
- 递归地考虑n除了p以外的其它质因数，也即n/p^k的所有质因数，它们共同贡献了$\mu(n/p^k)$，所以这个方法是正确的

莫比乌斯函数线性筛代码

```
miu[1]=1;
for (int i=2;i<=n;i++)
{
    if (!check[i])
    {
        prime[++top]=i;
        miu[i]=-1;
    }
    for (int j=1;j<=top&&i*prime[j]<=n;j++)
    {
        check[i*prime[j]]=1;
        if (i%prime[j]==0)
        {
            miu[i*prime[j]]=0;
            break;
        }
        else miu[i*prime[j]]=miu[i]*(-1);
    }
}
```

除数函数

- 准确地说，除数函数是一个函数族，在竞赛中，这是常用积性函数里最难的一类函数，要统一证明它们的积性较为复杂无聊，故我们在此略去（不过σ_0和σ_1的积性较为简单，大家可以自行尝试）
- 相应的，利用Euler筛法预处理除数函数的值也不容易，但是根据除数函数的定义，我们很容易得到一个利用Eratosthenes筛法的O(nloglognlogk)的算法预处理出σ_k的值，一般情况下这个复杂度已经可以接受了
- 至此，三类常用积性函数的线性筛法已经介绍完了，接下来简单看看其它比较平凡的积性函数的定义

其它积性函数

- 常函数(1)：$\forall n\in N$，$1(n)=1$
- 单位函数(id)：$\forall n\in N$，$id(n)=n$
- 幂函数(id_k)：$\forall n\in N$，$id_k(n)=n^k$
- 狄利克雷卷积单位函数(ε)：$\varepsilon(1)=1$，$\forall n>1$，$\varepsilon(n)=0$
- 元函数(e)：这是一个由命题映射到N的特殊函数，e[P]=1当且仅当P为真，否则e[P]=0
- 倒数函数(f)：$\forall n\in N$，$f(n)=1/n$
- 值得一提的是，这些函数都是完全积性函数

函数的积与Dirichlet卷积

- 设f、g是两个数论函数，则我们可以构造两个新的数论函数：
  - 乘积函数：$(f*g)(n)=f(n)g(n)$
  - Dirichlet卷积函数：$(f\times g)(n)=\sum_{d|n}f(d)g(\frac{n}{d})$
- 定理：若f、g都是积性函数，则两个新函数都是积性函数，若f、g都是完全积性函数，则两个新函数也都是完全积性函数
- 证明：将上述式子依次逐点展开即可得证

两个重要推论

- 推论1：$\mu \times 1 = \varepsilon$
- 证明：
  - 假设n=1，则显然$\mu \times 1(n)=1$
  - 假设$n \neq 1$，设n有r个质因数，因为有多重质因数时$\mu(d)=0$，故只需考虑所有的只含一重质因数的d，由二项式定理即可得证
- 推论2：$\varphi \times 1 = id$
- 证明：
  - 若n为质数，显然有$\varphi \times 1(n)=n$
  - 其它情况由$\varphi \times 1$和id都是积性函数即可得证

总结

- 至此，线性筛与积性函数就介绍完了
- 这部分内容比较枯燥，且所学的内容可能还不能直接应用到题目上，但还是希望大家能够认真钻研这些基本内容，以保证后续的反演学习能够顺利
- 牢记这几点：
  - 多考虑边界情况与特殊情形
  - 证明是为了记住定理与确信定理是正确的
  - 分清楚需要理解和需要记忆的要点

Thank you for listening!