Meissel-Lehmer Algorithm とその高速化について

tko919

2023 年 5 月 28 日

1 導入

$\pi(N)$、つまり N 以下の素数の個数を求める問題は、アルゴリズム的数論における古典的な問題として広く取り上げられている。N が小さい場合では Eratosthenes の篩などによって直接求めることができるが、今回はこの問題を解く有名なアルゴリズム (Meissel-Lehmer Algorithm) とその高速化について紹介する。

2 定義

$\pi(N) := N$ 以下の素数の個数　$p_i := i$ 番目の素数
$\phi(N, a) := N$ 以下の自然数のうち、最小の素因数が p_a より大きいものの個数
$P_k(N, a) := \phi(N, a)$ に含まれる自然数のうち、重複ありの素因数が丁度 k 個存在する (素因数分解の指数の和が k) ものの個数　但し、$P_0(N, a) = 1$ とする
$\delta(N) := N$ の最小素因数　$\theta(N) := N$ の最大素因数

$$\mu(N) := \begin{cases} (-1)^k & (N = p_{a_1} p_{a_2} \cdots p_{a_k}) \\ 0 & (d^2 | N) \end{cases}$$ 但し、$\mu(1) = 1$ とする

また、定義より直ちに

$$\pi(N) = P_1(N, a) + a \tag{1}$$

$$\sum_{k=0}^{\infty} P_k(N, a) = \phi(N, a) \tag{2}$$

$$\phi(N, a) = \phi(N, a-1) - \phi(N/p_a, a-1) \tag{3}$$

が認められる。

3 概要

(1) と (2) より、

$$\pi(N) = P_1(N, a) + a = a + \phi(N, a) - 1 - \sum_{k=2}^{\infty} P_k(N, a) \tag{4}$$

ここで $\phi(N, a)$ と $\displaystyle\sum_{k=2}^{\infty} P_k(N, a)$ を分けて計算することを考える。

- $\phi(N,a)$ の展開
  (3) を用いて再帰的に計算することで求められ、次の公式を得る。

$$\phi(N,a) = \sum_{\substack{m \le N \\ \theta(m) \le p_a}} \mu(m)[N/m] \tag{5}$$

- $\sum_{k=2}^{\infty} P_k(N,a)$ の計算
  $k=2$ の場合、片方の素数を固定して数え上げることで次の式変形を得る。

$$\begin{aligned} P_2(N,a) &= \#\{x \mid x \le N, x = p_b p_c, a < b \le c\} \\ &= \sum_{b=a+1}^{\pi(\sqrt{N})} \#\{c \mid b \le c \le \pi(N/p_b)\} \\ &= \sum_{b=a+1}^{\pi(\sqrt{N})} \pi(N/p_b) - (b-1) \\ &= \sum_{b=a+1}^{\pi(\sqrt{N})} \pi(N/p_b) + \binom{a}{2} - \binom{\pi(\sqrt{N})}{2} \end{aligned} \tag{6}$$

  よって、こちらも $\pi(M)$ を再帰的に計算することで求められる。また、$a = \pi(N^{1/3})$ と置くことで $P_k(N,a)(3 \le k)$ が全て 0 になる。a の計算はそのまま Eratosthenes の篩を使えばよい。

このアルゴリズムにより、計算量は線型のままだがかなり小さな定数倍を持って計算することが出来た。 [1]

4 高速化

以下の考察は Lagarias, Miller, Odlyzko らによる高速化に基づいている。基本的な方針としては、各パートで再帰的な処理を行っていたところをそのまま計算することである。
まず、(6) の $\sum_{b=a+1}^{\pi(\sqrt{N})} \pi(N/p_b)$ の計算について N/p_b は $N^{2/3}$ を超えないので、$N^{2/3}$ までの自然数を篩にかけてから順次足し合わせればよい。空間計算量を $N^{1/3}$ に抑えるために、$N^{2/3}$ までの自然数を $N^{1/3}$ 程度の区間に分けて処理することが出来る。また、篩にかける素数は高々 $N^{1/3}$ 程度なので区間の中に倍数が一つは存在し、この処理による計算量の増大は起きない。
次に、$\phi(N,a)$ の展開を高速化するために、$\phi(M,b)$ が以下のケースに入るとき再帰を終了する。

1. $b = 0 \wedge M \ge N^{2/3}$
2. $M < N^{2/3}$

ここで $M = N/x$ $(x|p_1p_2p_3\dots p_a)$ と置くと、$x \le N^{1/3}$ から 1 型の項数は $N^{1/3}$ を超えない。また、$\phi(M,b)$ の「親」は $\phi(M,b+1)$ または $\phi(Mp_{b+1},b+1)$ であり、前者は明らかに 2 型なので遷移は起きない。後者は $x/p_{b+1} \le N^{1/3}$ かつ p_{b+1} が x の最小素因数であるときに限り存在するので、2 型の項数は $\#\{m,b \mid m \le N^{1/3} \wedge p_{b+1} < \delta(m)\}$ より小さく、$N^{2/3}$ 以下であることが示された。

この打ち切りによって (5) から以下の式変形が得られる。$\phi(N,0)=[N]$ に注意されたい。

$$\phi(N,a)=\sum_{m\leq N^{1/3}}\mu(m)[N/m]-\sum_{p\leq N^{1/3}}\sum_{\substack{\delta(m)>p\\ m\leq N^{1/3}<mp}}\mu(m)\phi(N/mp,\pi(p)-1) \tag{7}$$

さて、前者の計算に必要な $\mu(m)$ の列挙は Eratosthenes の篩を応用することで解ける。具体的には、$N^{1/3}$ 以下の素数 p に対して、p の倍数に -1 倍して p^2 の倍数を 0 にする。後は $\mu(m)[N/m]$ を和に適宜加えれば $O(N^{1/3+\epsilon})$ で計算できる。

後者は愚直に $N^{2/3}$ 以下の自然数を篩にかけて二重和を計算すると線型時間かかってしまうが、これはパケット分割と Binary Indexed Tree (一点加算, 区間和を $O(\log N)$ で行うデータ構造) によって高速化することができる。具体的には、サイズ $N^{1/3}$ の区間 $[rN^{1/3},(r+1)N^{1/3})$ $(r<N^{1/3})$ に対して、N/mp が含まれるような m,p の組を全探索する。p を固定することで $m\leq N^{1/3}<mp$ を満たす m が半開区間になり探索範囲を狭めることができる。その範囲の中で $\delta(m)>p$ を満たすものを加算すればよい。しかし、区間ごとに累積和を取るとやはり線型かかってしまうので、各区間において $S[j]=\phi(rN^{1/3},j)$ $(j\leq a)$ と、区間内の自然数が篩で落とされたかどうかをフラグに持つサイズ $N^{1/3}$ の Binary Indexed Tree を管理して、p の昇順に更新する。$\delta(m)$ の列挙は前者の $\mu(m)$ と同様に、素数を検知したときにカウンターを増やし、p の倍数全てについてカウンターの値にすることで前計算が出来る。

計算量は $\phi(N,a)$ の二重和がボトルネックになり $O(N^{2/3+\epsilon})$ である。実際には他にも様々な高速化がなされている。 [2]

References

[1] ADITHYA C. GANESH, Efficient prime counting with the Meissel-Lehmer algorithm, December 2016, https://acgan.sh/posts/2016-12-23-prime-counting.html

[2] M. DELEGLISE AND J. RIVAT, COMPUTING $\pi(x)$: THE MEISSEL, LEHMER, LAGARIAS, MILLER, ODLYZKO METHOD, MATHEMATICS OF COMPUTATION Volume 65, Number 213 January 1996, 235-245, https://www.ams.org/journals/mcom/1996-65-213/S0025-5718-96-00674-6/S0025-5718-96-00674-6.pdf