Counting Primes

December 23, 2019

Abstract

Here we briefly detail the algorithms used within to evaluate $\pi(x)$ - the number of prime numbers less than x ...

1 The Maths

1.1 Sieve of Eratosthenes

This is the algorithm everyone knows - but it's worth detailing, because it feeds into the combinatorial algorithms. The sieve of Eratosthenes works by writing down a bit for each number from 2 to x. Then you walk along the sieve until you hit a bit at position p that's on. This number is a prime number, so you flip every bit that's a multiple of p. As a minor optimisation, all numbers less than p^2 will already have their final status determined, so you can start there and work up to x. Then, when you reach $\sqrt{x}$, you can stop and count - every number in the sieve will have its final status.

1.2 Legendre's Method

This works by thinking about the sieve of Eratosthenes. First, we define a function $\phi(x, a)$ to be the number of numbers at most x that are coprime to the first a prime numbers. Then, for the same reason that we can stop crossing off numbers at $\sqrt{x}$ (i.e. every between $\sqrt{x}$ and x is either prime or has a factor at most $\sqrt{x}$), we get that:

$$\pi(x) = \pi\left(\sqrt{x}\right) + \phi\left(x, \pi\left(\sqrt{x}\right)\right) - 1$$

Here, $\pi\left(\sqrt{x}\right)$ counts the number of prime numbers below $\sqrt{x}$, whereas $\phi\left(x, \pi\left(\sqrt{x}\right)\right)-$ 1 counts the number of primes between $\sqrt{x}$ and n. We need the minus 1 there as 1 is coprime to every number, so the ϕ term counts it as well.

1.2.1 Evaluation of ϕ

The reason this might be a good idea is that we have a recurrence we can use to evaluate ϕ. If p_a is the ath prime, then:

$$\phi(x, a) = \phi(x, a - 1) - \phi(x/p_a, a - 1)$$

We get this by thinking about numbers that are or aren't coprime to p_a below x. Many of the further methods either try to reduce the value of a in this evaluation, or reduce the depth at which we need to recurse.

1.3 The Meissel-Lehmer Method

Legendre's method by itself doesn't really help - we need an optimisation. We split the evaluation of ϕ into an infinite class of functions - define $P_k(x, a)$ to be the number of numbers at most x that have exactly k prime factors (not necessarily distinct) and that are coprime to the first a prime numbers. Then, clearly:

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

Notice that $P_0(x, a) = 1$ for $x \geq 1$. But also, provided that $a \leq \pi(x)$, we have (as P_1 counts terms with exactly 1 prime factor):

$$P_1(x, a) = \pi(x) - a$$

And hence, for $a \leq \pi(x)$ and $x \geq 1$ we have:

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

The reason that we transform the expression in this way, is that the minimum value of an integer that can contribute to the expression $P_k(x, a)$ is p_{a+1}^k. Hence, if $a \geq \pi\left(\sqrt[k]{x}\right)$, we have that $P_k(x, a) = 0$. This gives us our first practical methods for larger numbers - set $a = \pi\left(\sqrt[4]{x}\right)$ (Meissel's method) or set $a = \pi\left(\sqrt[3]{x}\right)$.

1.3.1 Meissel's Method

Chronologically, this method came first, but isn't the method that most other ones build off of. Perhaps this is because Meissel did not yet have the advantage of computers, so the advantage of evaluating a much smaller tree of ϕ was worth more than the simplicity of Lehmer's method.

Here we set $a = \pi \left(\sqrt[4]{x} \right)$

Evaluation of P_2

If $x = pq$ with p and q primes, then x/p is prime. For a term to contribute to $P_2(x, a)$, we must then have that $p, q > p_a$ and hence, by fixing p as the smaller prime, we want to evaluate:

$$P_2(x, a) = \sum_{b=a+1}^{\pi(\sqrt{x})} \left(\pi \left(x/p_b \right) - (b - 1) \right)$$

$$= \frac{a(a-1)}{2} - \frac{\pi \left(\sqrt{x} \right) \left(\pi \left(\sqrt{x} \right) - 1 \right)}{2} + \sum_{b=a+1}^{\pi(\sqrt{x})} \pi \left(x/p_b \right)$$

Evaluation of P_3

Similarly to P_2, we get that:

$$P_3(x, a) = \sum_{b=a+1}^{\pi(\sqrt[3]{x})} \sum_{c=b}^{\pi(\sqrt[3]{x})} \left(\pi \left(\frac{x}{p_b p_c} \right) - (c - 1) \right)$$

1.3.2 Lehmer's Method

Very similar to Meissel's method, except we set a to a larger value at $a = \pi \left(\sqrt[3]{x} \right)$. This has the benefit of making the calculation of P_2 slightly easier, and sets the P_3 term to 0. We don't use this method here, as it proves to be slower than Meissel's.

1.4 The Lagarias-Miller-Odlyzko Method

The LMO method introduces the ability to parallelise the computation, as well as reduce the size of the tree that gets computed. Here, we set $a =$

$$\pi\left(\sqrt[3]{x}\right)$$

1.4.1 Computation of P_2

This is as above. The computation is done in parallel by sieving blocks of the interval

1.5 The Delaglise-Rivat Method

1.6 The Gourdon Method

Fix values of y, z s.t. $x^{1/3} < y \leq z < x^{1/2}$ as in Delaglise-Rivat, and let k be a small (integer) constant. Let

$$\chi(\alpha) = \begin{cases} 2 \text{ if } \alpha < y \\ 1 \text{ if } \alpha \geq y \end{cases}$$

Let $\mu(n)$ be the Möbius function, $\gamma(n)$ the largest prime factor of n and $\delta(n)$ the smallest prime factor of n.

Furthermore, let $a = \pi(y)$, $b = \pi\left(x^{1/3}\right)$, $c = \pi\left((x/y)^{1/2}\right)$, $d = \pi(x^*)$ and

$x^* = \max\left(x^{1/4}, x/y^2\right)$. Then, define:

$$A = \sum_{x^* < p \leq x^{1/3}} \sum_{p < q \leq (x/p)^{1/2}} \chi\left(\frac{x}{pq}\right) \pi\left(\frac{x}{pq}\right)$$

$$B = \sum_{y < p \leq x^{1/2}} \pi\left(\frac{x}{p}\right)$$

$$C = -\sum_{p_k < p \leq x^*} \sum_{m:m \leq z < pm, \delta(m) > p, \gamma(m) \leq y, m > x/p^3} \mu(m)\left(\pi\left(\frac{x}{pm}\right) - \pi(p) + 2\right)$$

$$D = -\sum_{p_k < p \leq x^*} \sum_{m:m \leq z < pm, \delta(m) > p, \gamma(m) \leq y, m \leq x/p^3} \mu(m)\phi\left(\frac{x}{pm}, \pi(p) - 1\right)$$

$$\phi_0 = \sum_{n \leq y, \delta(n) > p_k} \mu(n)\phi\left(\frac{x}{n}, k\right)$$

$$\Sigma_0 = a - 1 + \frac{\pi\left(x^{1/2}\right)\left(\pi\left(x^{1/2}\right) - 1\right)}{2} - \frac{a(a-1)}{2}$$

$$\Sigma_1 = \frac{(a-b)(a-b-1)}{2}$$

$$\Sigma_2 = a\left(b - c - \frac{c(c-3)}{2} + \frac{d(d-3)}{2}\right)$$

$$\Sigma_3 = \frac{b(b-1)(2b-1)}{6} - b - \frac{d(d-1)(2d-1)}{6} + d$$

$$\Sigma_4 = \pi(y) \sum_{x^* < p \leq (x/y)^{1/2}} \pi\left(\frac{x}{py}\right)$$

$$\Sigma_5 = \sum_{(x/y)^{1/2} < p \leq x^{1/3}} \pi\left(\frac{x}{p^2}\right)$$

$$\Sigma_6 = -\sum_{x^* < p \leq x^{1/3}} \pi\left(\frac{x^{1/2}}{p^{1/2}}\right)^2$$

$$\Sigma = \Sigma_0 + \Sigma_1 + \Sigma_2 + \Sigma_3 + \Sigma_4 + \Sigma_5 + \Sigma_6$$

Then, we have that:

$$\pi(x) = A - B + C + D + \phi_0 + \Sigma$$

As is the case in the other methods, to make this optimal we pick $y =$

$Kx^{1/3} \log^3(x) \log \log(x)$, with K a parameter to be optimised around, but should be within an order of magnitude of 1 (I think). Further, we pick $z = Ly$ with $L > 1$ as a parameter to be optimised.

2 References

LMO: http://www.dtc.umn.edu/ odlyzko/doc/arch/meissel.lehmer.pdf DR: http://sweet.ua.pt/tos/bib/5.4.pdf Gourdon: http://numbers.computation.free.fr/Constants/P