adamant's blog

Dirichlet convolution. Part 1: Fast prefix sum computations

By **adamant**, history, 19 months ago,

Hi everyone!

Suppose that you need to compute some sum of a number-theoretic function that has something to do with divisors:

$$\sum_{k=1}^{n} \varphi(k) = ?$$

$$\sum_{k=1}^{n} \sum_{d|k} d^2 = ??$$

$$\sum_{x=1}^{n} \sum_{y=1}^{x} \gcd(x, y) = ?!?$$

As it turns out, such and many similar sums can be computed with Dirichlet convolution in $O(n^{2/3})$, and in this article we will learn how.

Let $f(n)$ and $g(n)$ be two arithmetic functions. Let $F(n)$ and $G(n)$ be their prefix sums, that is

$$F(n) = \sum_{i=1}^{n} f(i), \quad G(n) = \sum_{j=1}^{n} g(j).$$

We need to compute a prefix sum of the Dirichlet convolution $(f * g)(n)$. In this article, we will consider some general methods, and show how to do so in $O(n^{2/3})$ if we can compute prefix sums of $F(n)$ and $G(n)$ in all possible values of $\lfloor n/k \rfloor$ in this complexity.

- **Part 1: Fast prefix sum computation**
- **Part 2: Dirichlet series and prime counting**

ecnerwala previously mentioned that it is possible, but did not go into much detail. There is also a blog by **Nisiyama_Suzune**, which covers prefix sums of Dirichlet inverses in $O(n^{2/3})$.

Dirichlet convolution

Recall that the Dirichlet convolution of arithmetic functions $f(n)$ and $g(n)$ is defined as

$$\boxed{(f * g)(n) = \sum_{d|n} f(d) g\left(\frac{n}{d}\right) = \sum_{ij=n} f(i) g(j)}$$

Dirichlet convolution is often of interest in the context of multiplicative functions (i.e., functions such that $f(a)f(b) = f(ab)$ for $\gcd(a, b) = 1$), as the result can be proven to also be a multiplicative function. Moreover, quite often functions of interest can be expressed as a Dirichlet convolution. For example, the divisor function

$$\sigma_a(n) = \sum_{d|n} d^a$$

can be represented as a Dirichlet convolution of the constant function $f(n) = 1$ and the power function $g(n) = n^a$. There are also other similar identities, for example, for the Euler's totient function $\varphi(n)$, one can notice that

$$\sum_{d|n} \varphi(d) = n.$$

This is due to the fact that $\varphi\left(\frac{n}{d}\right)$ is the number of integers x such that $\gcd(x, \frac{n}{d}) = 1$, or, equivalently, $\gcd(x, n) = d$. This implies that $h(n) = n$ can be represented as the Dirichlet convolution of $f(n) = 1$ and $\varphi(n)$.

Hyperbola method

For now assume that we can compute any of f, g, F, G in $O(1)$. We can rewrite the prefix sum of the Dirichlet convolution as

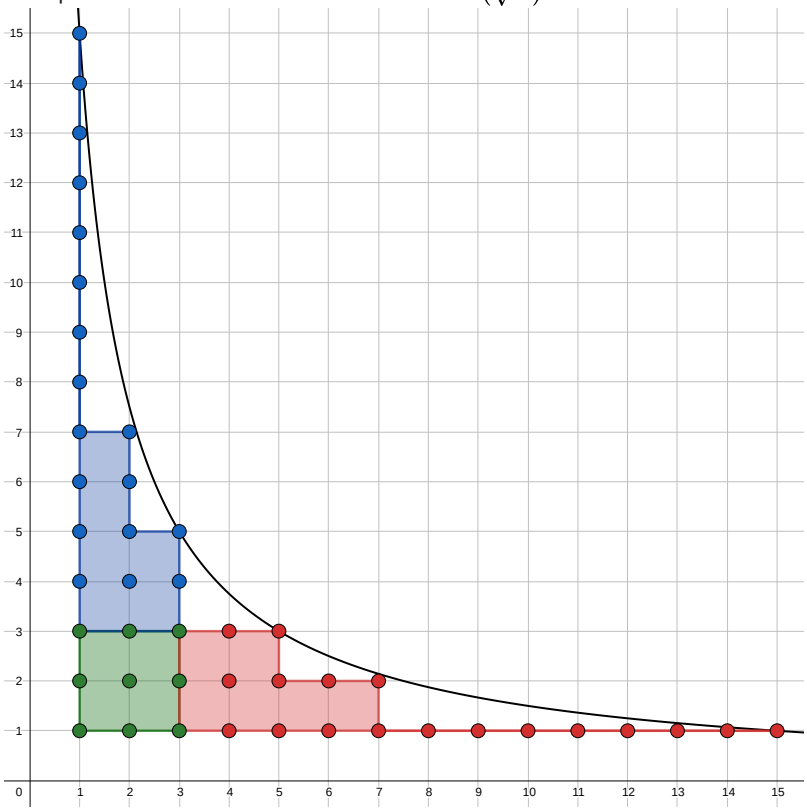
$$\sum_{t \le n} (f * g)(t) = \sum_{t \le n} \sum_{d \mid t} f(d) g\left(\frac{t}{d}\right) = \sum_{ij \le n} f(i) g(j).$$

In the expression above, at least one of i and j must be less than $\sqrt{n}$. This allows us to rewrite the expression as

$$\sum_{ij \le n} f(i) g(j) = \sum_{i=1}^{\lfloor \sqrt{n} \rfloor} f(i) \sum_{j=1}^{\lfloor n/i \rfloor} g(j) + \sum_{j=1}^{\lfloor \sqrt{n} \rfloor} g(j) \sum_{i=1}^{\lfloor n/j \rfloor} f(i) - \sum_{i=1}^{\lfloor \sqrt{n} \rfloor} f(i) \sum_{j=1}^{\lfloor \sqrt{n} \rfloor} g(j)$$

$$= \boxed{\sum_{i=1}^{\lfloor \sqrt{n} \rfloor} f(i) G(\lfloor n/i \rfloor) + \sum_{j=1}^{\lfloor \sqrt{n} \rfloor} g(j) F(\lfloor n/j \rfloor) - F(\lfloor \sqrt{n} \rfloor) G(\lfloor \sqrt{n} \rfloor)}$$

Thus, in this situation we can find a prefix sum of the Dirichlet convolution in $O(\sqrt{n})$.



Summation blocks in $ij \le 15$

Points in the green block are double-counted and should be subtracted

The representation of the prefix summation, as written above, is commonly known as the Dirichlet hyperbola method. Graphically, such summation splits lattice points below the $ij = n$ hyperbola into three parts (see the picture above). Besides exact computations, this method can also be used to get asymptotic estimations for summations like the one above.

Exercise 1: Solve **Project Euler — #401 Sum of Squares of Divisors**: Compute the prefix sum of $\sigma_2(n) = \sum_{d \mid n} d^2$ up to $n = 10^{15}$.

Choosing a better splitting point

Now, what if computing $F(i)$ and $G(j)$ requires vastly unbalanced amount of effort?

Example: Let's say that we want to compute to compute

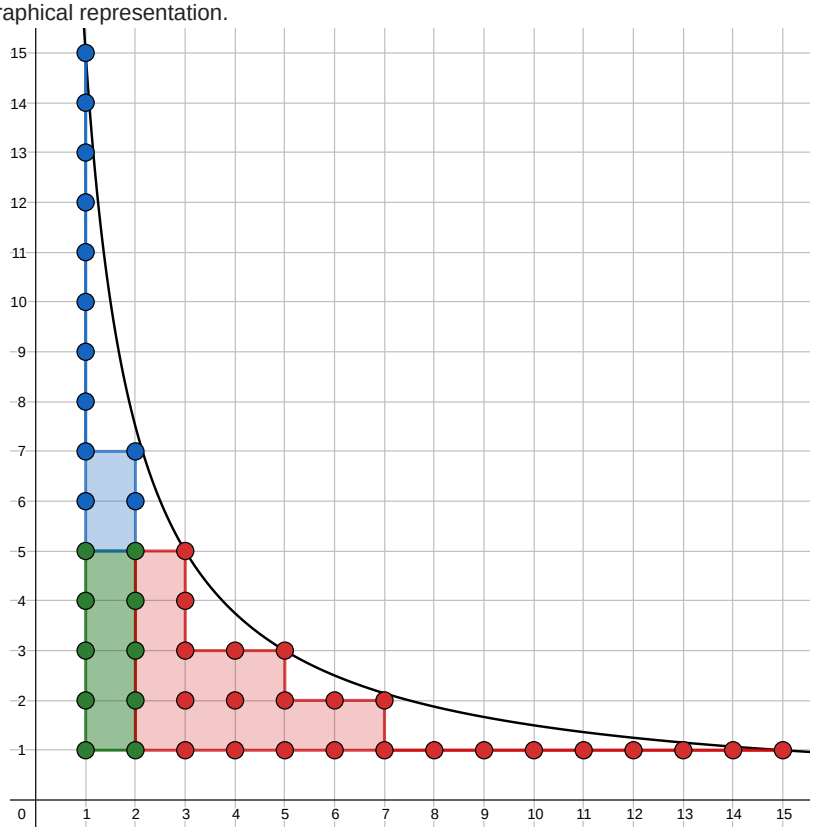
$$h(n) = \sum_{a \mid n} \sum_{b \mid a} b^2,$$

the sum of squares of divisors of all divisors of n. It can be represented as a Dirichlet convolution of $f(n) = 1$ and $g(n) = \sigma_2(n)$. For the first one, prefix sums can be computed in $O(1)$, and for the second one they can be computed in $O(\sqrt{n})$, see excercise 1.

In this case, we can bound the sums by a point (k, l) on the rectilinear convex hull of points under the hyperbola $kl = n$:

$$\sum_{ij \le n} f(i)g(j) = \sum_{i=1}^{k} f(i)G(\lfloor n/i \rfloor) + \sum_{j=1}^{l} g(j)F(\lfloor n/j \rfloor) - F(k)G(l)$$

See the picture below for a graphical representation.



For $n = 15$, we can choose $k = 2$ and $l = 5$ instead of $k = l = 3$

Let's now assume that we can compute $F(n)$ in n^α and $G(n)$ in n^β for $0 \le \alpha, \beta < 1$. What is the optimal splitting pair (k, l)?

Note that for (k, l) to be on the rectilinear convex hull of points below $kl = n$, it is necessary and sufficient to satisfy $kl \le n$ and $(k + 1)(l + 1) > n$, so we may essentially assume that $kl \sim n$. What is the time complexity required to compute the full sum? It is

$$\sum_{i=1}^{k} O\left(\frac{n^\beta}{i^\beta}\right) + \sum_{j=1}^{l} O\left(\frac{n^\alpha}{j^\alpha}\right) = O\left(\frac{n^\beta}{k^{\beta-1}} + \frac{n^\alpha}{l^{\alpha-1}}\right)$$

To minimize the total complexity, we should make the two summands equal. Substituting $l \sim n/k$, we get

$$n^\beta = nk^{\alpha+\beta-2} \implies \boxed{k = n^{\frac{1-\beta}{2-\alpha-\beta}}}$$

In particular, for $\alpha = 0$ and $\beta = 1/2$ we get $k = n^{1/3}$ and $l = n^{2/3}$, making for the total complexity $O(n^{2/3})$.

Adding precomputation

Okay now, but what if α and β are roughly the same, and also reasonably huge? For example $\alpha = \beta = 1/2$ or even $\alpha = \beta = 2/3$.

Due to symmetry, it's still optimal to split in $k = l = \lfloor \sqrt{n} \rfloor$, so we can't make it better by choosing a better splitting point. What we can do, however, is to precompute prefix sums of $f(n)$ and $g(n)$ up to some point $\sqrt{n} \le t < n$.

It's typically possible to do so with linear sieve (see here for details) in $O(t)$, after which the complexity for the full computation becomes

$$O(t) + \sum_{i=1}^{n/t} O(\tfrac{n^\alpha}{i^\alpha}) + O(\sqrt{n}) = O\left(t + \frac{n^\alpha}{n^{\alpha-1}/t^{\alpha-1}} + \sqrt{n}\right) = O\left(t + nt^{\alpha-1} + \sqrt{n}\right).$$

Optimizing for t makes first 2 summands equal, thus $t = n^{1/(2-\alpha)}$. This leads to $O(n^{2/3})$ for $\alpha = 1/2$ or $O(n^{3/4})$ for $\alpha = 2/3$.

Adding even more precomputation

The result above is still not quite satisfactory. We see that $n^{2/3}$ is a kind of magical bound, to which a lot of things reduce. Given that, it would be very nice to stick to it, that is to somehow guarantee that if we can compute prefix sums of $f(n)$ and $g(n)$ in $O(n^{2/3})$, then we also can compute prefix sums of $(f * g)(n)$ in $O(n^{2/3})$. Turns out that it is possible to do! We just need a bit stronger guarantees.

Claim: Let $f(n)$ and $g(n)$ be such that $F(\lfloor n/k \rfloor)$ and $G(\lfloor n/k \rfloor)$ are known for all possible arguments. Then we can compute prefix sum $H(n)$ of $h(n) = (f * g)(n)$ in $O(\sqrt{n})$. Moreover, we can find $H(\lfloor n/k \rfloor)$ for all possible arguments in $O(n^{2/3})$.

Note 1: There might only be at most $2\sqrt{n}$ distinct values of $\lfloor n/k \rfloor$, because either k or $\lfloor n/k \rfloor$ must be smaller than $\sqrt{n}$.

Note 2: The result above allows us to repeatedly compute Dirichlet convolutions without blowing the complexity, as it will pin to $O(n^{2/3})$.

Proof: The result about $H(n)$ immediately follows from the hyperbola method.

Additionally, using the identities $\left\lfloor \frac{\lfloor n/a \rfloor}{b} \right\rfloor = \lfloor \frac{n}{ab} \rfloor$ and $\left\lfloor \sqrt{\left\lfloor \frac{n}{k} \right\rfloor} \right\rfloor = \left\lfloor \sqrt{\frac{n}{k}} \right\rfloor$, we can compute $H(\lfloor n/k \rfloor)$ with the hyperbola method as

$$\boxed{H(\lfloor n/k \rfloor) = \sum_{i=1}^{\left\lfloor \sqrt{n/k} \right\rfloor} f(i) G\left(\left\lfloor \frac{n}{ki} \right\rfloor\right) + \sum_{j=1}^{\left\lfloor \sqrt{n/k} \right\rfloor} g(j) F\left(\left\lfloor \frac{n}{kj} \right\rfloor\right) - F\left(\left\lfloor \sqrt{\frac{n}{k}} \right\rfloor\right) G\left(\left\lfloor \sqrt{\frac{n}{k}} \right\rfloor\right)}$$

in $O\left(\sqrt{\frac{n}{k}}\right)$, as any individual summand is computed in $O(1)$ after pre-computation.

Note that $\lfloor \sqrt{t} \rfloor$ always lies in the corner of the rectilinear convex hull of points below $kl = t$. There are two cases two consider here:

1. The corner is "concave", that is $\lfloor \sqrt{t} \rfloor (\lfloor \sqrt{t} \rfloor + 1) \le t$. It means that $\lfloor \sqrt{t} \rfloor = \lfloor t/l \rfloor$ for $\lfloor \sqrt{t} \rfloor < l \le \left\lfloor \frac{t}{\lfloor \sqrt{t} \rfloor} \right\rfloor$.
2. The corner is "convex", that is $\lfloor \sqrt{t} \rfloor (\lfloor \sqrt{t} \rfloor + 1) > t$. It means that $\lfloor \sqrt{t} \rfloor = \lfloor t/l \rfloor$ for $\left\lfloor \frac{t}{\lfloor \sqrt{t} \rfloor + 1} \right\rfloor < l \le \lfloor \sqrt{t} \rfloor$.

Note 3: Graphically, on the picture above the corner $(3, 3)$ is "concave", and the corner $(3, 5)$ is "convex".

In either case, it means that $\lfloor \sqrt{t} \rfloor$ can always be represented as $\lfloor t/l \rfloor$ for some l. Applying this result to $\lfloor n/k \rfloor$ instead of t, we deduce that $\left\lfloor \sqrt{\frac{n}{k}} \right\rfloor$ can be represented as $\left\lfloor \frac{\lfloor n/k \rfloor}{l} \right\rfloor = \lfloor \frac{n}{kl} \rfloor$, thus the values of $F(n)$ and $G(n)$ in $\left\lfloor \sqrt{\frac{n}{k}} \right\rfloor$ must also be known, and do not contribute any extra burden to the computation time.

Thus, the overall complexity is the sum of $\sqrt{\frac{n}{k}}$ over all values of $\lfloor n/k \rfloor$, in which we compute it with the hyperbola method. Since we can pre-compute these values up to $n/k \sim n^{2/3}$ in $O(n^{2/3})$ with the linear sieve, we're only interested in values of k up to $n^{1/3}$, which takes

$$\sum_{k=1}^{\sqrt[3]{n}} \sqrt{\frac{n}{k}} \sim \int_0^{\sqrt[3]{n}} \sqrt{\frac{n}{x}} dx = 2n^{2/3},$$

proving the overall complexity of $O(n^{2/3})$.

Dirichlet inverse

Example: **Library Judge — Sum of Totient Function**. Find the sum of $\varphi(n)$ up to $n \le 10^{10}$.

Solution: Explained below, see submission for implementation details.

The result above provides us with an efficient method of computing the prefix sums of the convolution of two sequences. However, it often happens that we can compute the prefix sums of the convolution $(f * g)(n)$, and of one of the functions $f(n)$, but not of the other.

For example, consider the prefix sums of the Euler's totient function $\varphi(n)$. We already know that the Dirichlet convolution of $\varphi(n)$ and $f(n) = 1$ is $h(n) = n$. We can easily compute prefix sums of $f(n)$ and $h(n)$. But computing them for $\varphi(n)$ is much more difficult.

What we can do here is consider the Dirichlet inverse of $f(n) = 1$. Dirichlet inverse of a function $f(n)$ is the function $f^{-1}(n)$ such that $(f * f^{-1})(n)$ equates to the Dirichlet neutral function, which is defined as 1 for $n = 1$ and to 0 otherwise. As follows from its name, Dirichlet convolution of any function with the Dirichlet neutral yields the initial function.

From Möbius inversion formula, the inverse for it is known to be $\mu(n)$, the Möbius function, hence we can represent $\varphi(n)$ as

$$\varphi(n) = \sum_{d|n} d\mu\left(\frac{n}{d}\right),$$

that is as the Dirichlet convolution of $f(n) = n$ and $\mu(n)$. This reduces the task of finding prefix sums for $\varphi(n)$ to finding prefix sums of the Dirichlet convolution of $f(n) = n$ and $\mu(n)$. But how do we find the prefix sums of $\mu(n)$ (also known as the Mertens function)?

Claim: Let $f(n)$ be such that we can find values of $F\left(\lfloor n/k \rfloor\right)$ for all possible arguments in $O(n^{2/3})$. Then we can also find the prefix sums for all possible arguments of the Dirichlet inverse $f^{-1}(n)$ in $O(n^{2/3})$.

Proof: Let $F^{-1}(n)$ be the prefix sum of $f^{-1}(n)$ up to n. The hyperbola formula allows us to compute $F^{-1}(n)$ in $O(\sqrt{n})$, granted that we already know all values of $F\left(\lfloor n/k \rfloor\right)$ and $F^{-1}\left(\lfloor n/k \rfloor\right)$:

$$F^{-1}(n) = \frac{1}{f(1)}\left[1 + F(\lfloor\sqrt{n}\rfloor)F^{-1}(\lfloor\sqrt{n}\rfloor) - \sum_{i=1}^{\lfloor\sqrt{n}\rfloor} f^{-1}(i)F(\lfloor n/i \rfloor) - \sum_{i=2}^{\lfloor\sqrt{n}\rfloor} f(i)F^{-1}(\lfloor n/i \rfloor)\right]$$

Note: For non-zero multiplicative functions, $f(1) = 1$.

With this in mind, we can compute all values of $F^{-1}(n)$ up to $O(n^{2/3})$ with linear sieve, and then compute $F^{-1}\left(\lfloor n/k \rfloor\right)$ one by one in the increasing order of arguments.

Each value is computed in $O\left(\sqrt{\frac{n}{k}}\right)$ starting with $k \approx \sqrt[3]{n}$ and going up to $k = 1$. This gives the $O(n^{2/3})$ algorithm to find the Dirichlet inverse, and its analysis is essentially the same as with the algorithm for the Dirichlet convolution.

As a proof of concept, I have implemented the code to compute the sums of $\mu(n)$ and $\varphi(n)$ up to a given bound:
code

You can use $M(10^{12}) = 62366$ and $\Phi(10^{12}) = 303963550927059804025910$ to check your implementation.

Exercise 2: Solve **Project Euler — #625 Gcd Sum**: Compute $G(n) = \sum_{j=1}^{n}\sum_{i=1}^{j} \gcd(i, j)$ for $n = 10^{11}$.

Exercise 3: Solve **Project Euler — #351 Hexagonal Orchards**.

Further reading

- **[Tutorial] Math note — linear sieve** — how to compute prefix of multiplicative functions up to n in $O(n)$ with linear sieve.
- **[Tutorial] Math note — Möbius inversion** — how to apply Möbius inversion to some number-theoretic problems.
- **[Tutorial] Math note — Dirichlet convolution** — another entry on Dirichlet convolution inversion in $O(n^{2/3})$.
- **Summing Multiplicative Functions (Pt. 1)** — more on computing prefix sums of multiplicative functions.

UPD: Turns out, it is also possible to compute prefix sums in $\lfloor n/k \rfloor$ for Dirichlet convolution and Dirichlet inverse in $O(n^{2/3})$ while only using $O(\sqrt{n})$ memory, and without linear sieving:
code

See the comment below for details.

tutorial, dirichlet convolution