Computing the (number or sum of) inverses of Euler's totient and other multiplicative functions

Max A. Alekseyev

George Washington University, Washington, DC, U.S.A.

Abstract. We propose a generic algorithm for computing the inverses of a multiplicative function. We illustrate our algorithm with Euler's totient function and the sum of k-th powers of divisors. Our approach can be further adapted for computing certain functions of the inverses, such as their quantity, sum, or the smallest/largest inverse, which may be computed without and possibly faster than the inverses themselves.

1 Introduction

A value of multiplicative function f on a positive integer n equals the product of its values on the prime powers in the prime factorization of n. That is, if $n = p_1^{e_1} \cdot p_2^{e_2} \cdots p_m^{e_m}$ where $p_1 < p_2 < \cdots < p_m$ are primes, then

$$f(n) = \prod_{i=1}^{m} f(p_i^{e_i}).$$

In particular, $f(1) = 1$.

Famous examples of multiplicative functions include $\tau(n)$, the number of divisors of n (with $\tau(p^e) = e + 1$); $\sigma_k(n)$, the sum of k-th powers of divisors of n (with $\sigma_k(p^e) = \frac{p^{k(e+1)}-1}{p^k-1}$); and Euler's totient function $\varphi(n)$ (with $\varphi(p^e) = (p-1) \cdot p^{e-1}$)).

In our work, we propose a generic algorithm for computing the set of inverses (full pre-image) $f^{-1}(n)$ of a multiplicative function f for a given integer n with a known factorization (which otherwise may be a bottleneck to obtain; e.g., Contini et al. [2] proved that computing $\varphi^{-1}(n)$ is as hard as factorization of n). While computing inverses of Euler's totient function was studied to some extent [3,2,1], computing inverses of other multiplicative functions (with a notable and simple exception of $\tau(n)$), to the best of our knowledge, was not addressed in the literature. Our algorithm may be viewed as a generalization and streamlining of the "intelligent exhaustive search" for $\varphi^{-1}(n)$ in [2]. We present an underlying idea of the algorithm in the elegant form of formal Dirichlet series and extend it to computing certain functions of the inverses, without computing the whole set of inverses.

2 Formal Dirichlet series formulae

From now on, we assume that f is a fixed multiplicative function.

We find it convenient to define binary multiplication $\times$ and addition $+$ on sets of positive integers as follows: $U \times V = \{u \cdot v \,:\, u \in U, v \in V\}$ and $U + V = U \cup V$. Equipped with these operations the set $\mathcal{P}_{\text{fin}}(\mathbb{Z}_{>0})$ of finite nonempty subsets of positive integers forms a commutative semiring (with the multiplicative identity $\{1\}$) and allows us to consider Dirichlet series with coefficients from this semiring.

Theorem 1. *We have the following identity for Dirichlet series of variable s over the semiring $(\mathcal{P}_{fin}(\mathbb{Z}_{>0}), \times, +)$:*

$$\sum_{n\geq 0} \frac{f^{-1}(n)}{n^s} = \bigtimes_{\text{prime } p} \quad \sum_{e=1}^{\infty} \frac{\{p^e\}}{f(p^e)^s}. \tag{1}$$

For a fixed integer $n > 0$ and every $d \mid n$, we further have:

$$f^{-1}(d) = \text{Coeff}_{d^{-s}} \bigtimes_{\text{prime } p} \quad \sum_{e:\, f(p^e)\mid n} \frac{\{p^e\}}{f(p^e)^s}. \tag{2}$$

Proof. Multiplicativity of f implies that if $n = f(m)$ for some positive integers m, n and m has factorization $m = p_1^{e_1} \cdot p_2^{e_2} \cdots p_k^{e_k}$ where $p_1 < p_2 < \cdots < p_k$ are primes, then $n = f(p_1^{e_1}) \cdot f(p_2^{e_2}) \cdots f(p_k^{e_k})$. It follows that n is the product of factors of the form $f(p^e)$, where p is prime and e is a positive integer and no two factors share the same p. In other words,

$$f^{-1}(n) = \sum_{f(p_1^{e_1})\cdots f(p_k^{e_k})=n} \bigtimes_{i=1}^{k} \{p_i^{e_i}\}, \tag{3}$$

where the sum is taken over various vectors of primes $p_1 < p_2 < \cdots < p_k$ and various positive integer exponents $e_1, e_2, \ldots, e_k$ that satisfy $f(p_1^{e_1}) \cdot f(p_2^{e_2}) \cdots f(p_k^{e_k}) = n$. Multiplying (3) by n^{-s}, we get

$$\frac{f^{-1}(n)}{n^s} = \sum_{f(p_1^{e_1})\cdots f(p_k^{e_k})=n} \bigtimes_{i=1}^{k} \frac{\{p_i^{e_i}\}}{f(p_i^{e_i})^s}.$$

Summing over $n \geq 0$, we obtain (1).

We remark that for a prime power p^e, $f(p^e)$ may participate in a factorization of n only if $f(p^e) \mid n$. Hence, to obtain the full pre-image $f^{-1}(n)$ from (1) for a given n, we can restrict our attention only to such prime powers:

$$f^{-1}(n) = \text{Coeff}_{n^{-s}} \bigtimes_{\text{prime } p} \quad \sum_{e:\, f(p^e)\mid n} \frac{\{p^e\}}{f(p^e)^s}. \tag{4}$$

We further remark that for every $d \mid n$, the coefficients of d^{-s} in the series in the right hand side of (4) and (1) coincide, which implies formula (2). □

Let $(X, \otimes, \oplus)$ be a commutative semiring. We call a mapping $C : (\mathcal{P}_{\text{fin}}(\mathbb{Z}_{>0}), \times, +) \to (X, \otimes, \oplus)$ a *weak homomorphism* if for any $U, V \in \mathcal{P}_{\text{fin}}(\mathbb{Z}_{>0})$, we have $C(U \times V) = C(U) \otimes C(V)$ whenever the sets U and V are element-wise *coprime* (i.e., $\gcd(u, v) = 1$ for any $u \in U$ and $v \in V$) and $C(U + V) = C(U) \oplus C(V)$ whenever U, V are disjoint. It is easy to see that if C is a homomorphism, then it is also a weak homomorphism.

Theorem 2. *Let $(X, \otimes, \oplus)$ be a commutative semiring and $C : (\mathcal{P}_{fin}(\mathbb{Z}_{>0}), \times, +) \to (X, \otimes, \oplus)$ be a weak homomorphism, then*

$$\bigoplus_{n \geq 0} \frac{C(f^{-1}(n))}{n^s} = \bigotimes_{\text{prime } p} \bigoplus_{e=1}^{\infty} \frac{C(\{p^e\})}{f(p^e)^s} \tag{5}$$

and for a fixed positive integer n and every $d \mid n$,

$$C(f^{-1}(d)) = \text{Coeff}_{d^{-s}} \bigotimes_{\text{prime } p} \bigoplus_{e:\, f(p^e) \mid n} \frac{C(\{p^e\})}{f(p^e)^s}. \tag{6}$$

Proof. We remark that the sets inside the product in (3) are coprime, while the products inside the sum are disjoint. Since C is a weak homomorphism, we have

$$C(f^{-1}(n)) = \sum_{f(p_1^{e_1}) \cdots f(p_k^{e_k}) = n} \bigtimes_{i=1}^{k} C(\{p_i^{e_i}\}),$$

which further implies identity (5).

Formula (6) is derived from (5) by the same arguments we used to derive (2) from (1). □

The formula (6) under appropriate choice of the homomorphism C and its codomain $(X, \otimes, \oplus)$ allows us to efficiently compute certain functions of the pre-images without computing the pre-images themselves. In the next section we give some particular examples.

3 Examples of weak homomorphisms

Our first, rather trivial example is given by $(X, \otimes, \oplus) = (\mathcal{P}_{\text{fin}}(\mathbb{Z}_{>0}), \times, +)$ with C being the identity homomorphism. In this case, formulae (5) and (6) simply represent the original formulae (1) and (2) for the full pre-images. We will still keep this example in mind to fit computation of the full pre-images into our generic algorithm.

Our second example is given by $(X, \otimes, \oplus) = (\mathbb{Z}_{>0}, \cdot, \max)$, which is a commutative semiring of positive integers with the standard integer multiplication and a binary maximum operation (i.e., $u \oplus v = \max\{u, v\}$). The mapping $C(U) = \max U$, giving the maximum element of a set, represents a homomorphism between $(\mathcal{P}_{\text{fin}}(\mathbb{Z}_{>0}), \times, +)$ and $(\mathbb{Z}_{>0}, \cdot, \max)$.

Similarly, the mapping $C(U) = \min U$ represents a homomorphism between $(\mathcal{P}_{\text{fin}}(\mathbb{Z}_{>0}), \times, +)$ and the commutative semiring $(\mathbb{Z}_{>0}, \cdot, \min)$.

An example of a weak homomorphism, which is not a homomorphism, is given by $(X, \otimes, \oplus) = (\mathbb{Z}_{\geq 0}, \cdot, +)$, i.e., semiring of nonnegative integers with the standard integer multiplication and addition, and $C_q(U) = \sum_{u \in U} u^q$, where $q \geq 0$ is a fixed integer. In particular, $C_0(U) = |U|$ represents the cardinality of a set U, while $C_1(U)$ is the sum of elements of U.

4 Algorithm for computing $C(f^{-1}(n))$

In addition to a multiplicative function f, we now fix a commutative semiring $(X, \otimes, \oplus)$ and a weak homomorphism $C : (\mathcal{P}_{\text{fin}}(\mathbb{Z}_{>0}), \times, +) \to (X, \otimes, \oplus)$. For a given integer n (and its prime factorization), computing $C(f^{-1}(n))$ naturally splits into three major steps.

First, from the known prime factorization of n, we easily compute the set of its divisors $D = \{d_1, d_2, \ldots, d_k\}$, where $k = \tau(n)$.

Second, we iteratively compute the right hand side of (6) restricted to the terms with denominators d^s for $d \in D$. Namely, we compute the *atomic series*

$$L_p = \bigoplus_{e:\ f(p^e) \mid n} \frac{C(\{p^e\})}{f(p^e)^s} = \bigoplus_{d \in D} \frac{A_d}{d^s}$$

for every prime p that possesses at least one integer $e > 0$ with $f(p^e) \mid n$.[1] Here

$$A_d = \bigoplus_{e:\ f(p^e) = d} C(\{p^e\}). \tag{7}$$

Internally we represent each such series L_p as an associative array $d \mapsto A_d$ for $d \in D$.

Third, we multiply the constructed atomic series $L_{p_1}, L_{p_2}, \ldots, L_{p_\ell}$ and compute partial products $P_0 = \frac{C(\{1\})}{1^s}$, $P_1 = P_0 \otimes_D L_{p_1}$, …, $P_\ell = P_{\ell-1} \otimes_D L_{p_\ell}$, where $\otimes_D$ denotes restriction of the result of $\otimes$ to the terms with denominators from D. Each multiplication is computed by the formula:

$$\left(\bigoplus_{d \in D} \frac{B_d}{d^s} \right) \otimes_D \left(\bigoplus_{d \in D} \frac{A_d}{d^s} \right) = \bigoplus_{d \in D} \frac{\bigoplus_{t \mid d} B_t \otimes A_{d/t}}{d^s}.$$

That is, if the associative array $d \mapsto B_d$ represents the partial product P_j then we replace it with the associative array $d \mapsto \bigoplus_{t \mid d} B_t \otimes A_{d/t}$ representing the partial product P_{j+1}. This can be done in-place by computing new coefficients for d going over the elements of D in decreasing order.

The coefficient of n^{-s} in the final product P_ℓ gives us $C(f^{-1}(n))$.

[1] We remark that if there is no such $e > 0$, then $L_p = \frac{C(\{1\})}{1^s}$ represents the identity for $\otimes$-multiplication of Dirichlet series.

We remark that the second step of the algorithm is specific to particular function f. We will illustrate this step with some examples in the next section. Below we briefly analyze space and time requirements for the generic third step.

The space complexity is proportional to $\tau(n)$ times the maximum size of $C(f^{-1}(d))$ for $d \mid n$. For C giving size, sum, maximum or minimum elements, under the assumption that the size of $f(x)$ is not significantly smaller than the size of x, the space complexity becomes simply $O(\tau(n) \cdot \log n)$.

The time complexity is bounded by $O(\ell \cdot \tau(n))$ $\otimes$-multiplications and $\oplus$-additions of values of C (assuming constant time for associative array queries). E.g., in computing the size/maximum/minimum of Euler's totient function inverses, we have $\ell \leq \tau(n)$ and thus $O(\tau(n)^2)$ multiplications and additions of integers of size $O(\log(n))$, which can be done in $O(\tau(n)^2 \cdot \log(n)^2)$ time.

5 Examples

We remark that our algorithm is generic and works for any multiplicative function f, provided that we can construct atomic series $L_{p_1}, \ldots, L_{p_\ell}$. Namely, we need to determine suitable p and compute the corresponding coefficients A_d defined by (7). Below we give specific examples.

5.1 Euler's totient function, φ

Since for $e > 0$, $\varphi(p^e) = (p-1)p^{e-1}$, $\varphi(p^e) \mid n$ implies that $p - 1$ divides d and $e \leq \nu_p(d) + 1$, where $\nu_p(d)$ is the p-adic valuation of d, i.e., the maximum integer t such that $p^t \mid d$ but $p^{t+1} \nmid d$. So we need to compute the set $S = \{p : p - 1 \in D \text{ and } p \text{ is prime}\}$, which can be done by going over the elements d of D and testing if $p = d + 1$ is prime. The set S gives us the indicies of atomic series. For every prime $p \in S$, we compute the corresponding atomic series:

$$L_p = \frac{C(\{1\})}{1^s} \oplus \bigoplus_{e=1}^{\nu_p(n)+1} \frac{C(\{p^e\})}{((p-1)p^{e-1})^s}.$$

5.2 Sum of k-th powers of divisors, σ_k

Suppose that $\sigma_k(p^e) = d$ for some $d \in D$. Since $\sigma_k(p^e) = 1 + p + p^2 + \cdots + p^e$, we have $p^e < d \leq (1+p)^e$, implying that $p = \lfloor (d-1)^{1/e} \rfloor$.

So to find p^e such that $\sigma_k(p^e) \mid n$, we let d run over the divisors of n and e run incrementally from 1 to $\lfloor \log_2(d-1) \rfloor$. For each such pair d, e, we test if $p = \lfloor \log_2(d-1) \rfloor$ is prime and if so, add term $\frac{C(\{p^e\})}{d^s}$ to L_p. Here we assume that initially all $L_p = \frac{C(\{1\})}{1^s}$ and only those L_p that were enriched with additional terms in the above process represent the atomic series.

5.3 Sequences in the OEIS

The Online Encyclopedia of Integer Sequences [4] contains a number of sequences, for which the proposed algorithm allows one to compute many terms:[2]

	$\varphi^{-1}(n!)$	$\sigma^{-1}(n!)$	$\varphi^{-1}(10^n)$	$\sigma^{-1}(10^n)$	$\sigma^{-1}(p_n\#)$
size	A055506	A055486	A072074	A110078	A153078
min	A055487	A055488	A072075	A110077	A153076
max	A165774	A055489	A072076	A110076	A153077

References

1. Coleman, R.: On the image of Euler's totient function. arXiv preprint arXiv:0910.2223 (2009)
2. Contini, S., Croot, E., Shparlinski, I.: Complexity of inverting the Euler function. Mathematics of computation 75(254), 983–996 (2006)
3. Gupta, H.: Euler's totient function and its inverse. Indian Journal of Pure and Applied Mathematics 12(1), 22–30 (1981)
4. The OEIS Foundation: The On-Line Encyclopedia of Integer Sequences. Published electronically at `http://oeis.org` (2014)

[2] Some of the current records in the number of computed terms belong to Ray Chandler.