

オイラー関数の多重合成を計算 するアルゴリズム

梶田光

内容

3つのアイデアの(ポスターや論文で伝えきれない)“気持ち”について

1. 部分分解とオイラー関数
2. $k = 2$ の場合: 調和級数のオーダーと空間計算量の見積り
3. 一般の場合: primechain の繋げ方

1. 部分分解とオイラー関数

部分分解は競技プログラミングでよく使われる“平方分割”のアイデアから.

- 素因数分解をすべて書き下して計算するのは時間がかかる
- 一般の n について $\varphi(n)$ を配列から取得するには必要な空間が大きすぎる

\sqrt{N} ギリギリまで素因数の積を順番に f_0, f_1 に詰め込むというアルゴリズムが時間と空間のトレードオフを現実的な範囲に落とし込む.

2. $k = 2$ の場合: 調和級数のオーダーと空間計算量の見積り

$k = 2$ のケースから考えたのは初等整数論の研究の方でまず $\varphi^2(n)$ について考えていたから.

n が (\sqrt{N} より) 大きな素数の倍数であった場合の $\varphi^2(n)$ の処理が難しいが,

調和級数 $H_n = \sum_{k=1}^n \frac{1}{k} \sim \log n$ を考えると \sqrt{N} 程度の長さの区間 $[\text{start}, \text{end}]$ の中の n の

\sqrt{N} より大きい素因数が含まれている可能性のある区間 $\left[\frac{\text{start}}{2}, \frac{\text{end}}{2}\right], \left[\frac{\text{start}}{3}, \frac{\text{end}}{3}\right], \dots, \left[\frac{\text{start}}{\sqrt{N}}, \frac{\text{end}}{\sqrt{N}}\right]$ をすべてメモリにマップしてしまっても空間計算量は $O(\sqrt{N} \log N)$ と小さい. (harmonic map)

3. 一般の場合: primechain の繋げ方

一般の k では,

- $f_2(n) > \sqrt{N}$ ならば $f_0(f_2(n) - 1), f_1(f_2(n) - 1)$
- さらに $f_2(f_2(n) - 1) > \sqrt{N}$ ならば $f_0(f_2(f_2(n) - 1) - 1), f_1(f_2(f_2(n) - 1) - 1)$
- さらに $f_2(f_2(f_2(n) - 1) - 1) > \sqrt{N}$ ならば $f_0(f_2(f_2(f_2(n) - 1) - 1) - 1), f_1(f_2(f_2(f_2(n) - 1) - 1) - 1)$
- \vdots

という入れ子の数列を最大 $k - 1$ まで取得しなければ $\varphi^k(n)$ が計算できないが, ありうる範囲を直接すべてたどって $O(k\sqrt{N}\log^k N)$ という巨大な量のメモリを消費することはできない.

=> 数列 $f_2(n), f_2(f_2(n) - 1), f_2(f_2(f_2(n) - 1) - 1), \dots$ はすべて harmonic map の範囲内. (GPT-5 との議論から発展.)

=> ディスク容量 $O(kN)$, 空間計算量 $O(\sqrt{N}\log N)$ で計算ができる.

詳細な議論



<https://github.com/hikaru-kajita/mathematics/tree/main/multiphi-computation>

Thank you!