素数の数え上げと乗法的関数の和

えびちゃん (rsk0315)

Jun. 29, 2022 @ ねこねこ勉強ぱーてぃ

更新：Jul. 2 00:56, 2022 (7e4c096)

目標

以下のことがわかるようになる：

- $\mathbb{N} \cap [1, n]$ の素数の個数 $\pi(n)$ や k 乗和[1] を求める
  - in $O(n^{3/4} / \log(n))$ time
  - in $O(n^{2/3})$ time
  - in $O(n^{2/3} / \log(n)^{1/3})$ time
- 乗法的関数 f について $\sum_{i=1}^{n} f(i)$ を求める
  - in $O(n^{3/4} / \log(n))$ time? ← 解析は未解決
  - in $O(n^{2/3})$ time

[1] k は定数とする。

記法に関して I

任意の二項演算 $\circ : S \times T \to S$ と $(x, y) \in S \times T$ に対して、

$$x \overset{\circ}{\leftarrow} y$$

で、$x \leftarrow x \circ y$ を表すものとする。`x += y` のような気持ち[2]。

特に、今回の内容においては、$x \leftarrow x - (y - z)$ の括弧を省いて $x \overset{-}{\leftarrow} y - z$ と書けるのがうれしい。

[2] LaTeX で $x += y$ などと書くのは見栄えが悪くて好きではない。

記法に関して II

擬似コード中において、ループ順が重要なときは列の形で

$$\textbf{foreach } i \leftarrow (1, \ldots, n) \textbf{ do}$$

と書き、そうでないときは集合の形で

$$\textbf{foreach } i \in \{1, \ldots, n\} \textbf{ do}$$

と書いている。

変数への代入には $v \leftarrow a$ を用いるが、定数の宣言のときには $v = a$ を用いることもある。

まずは愚直から I

Algorithm 2.1: 愚直に数え上げ

```
function PRIMECOUNT-NAÏVE(n)
    π ← 0
    foreach i ∈ {2, . . . , n} do
        if i is prime then          ▷ 試し割り法で判定
            π ←+ 1
    return π
```

これは $\Theta(n^{3/2}/\log(n))$ 時間。

$1/\log(n)$ は、各試し割りに必要な回数の解析に基づく[3]。

[3] https://twitter.com/259_Momone/status/1443890427514351622 など。

まずは愚直から II

Algorithm 2.2: 篩で数え上げ

```
function PRIMECOUNT-SIEVE(n)
    π ← 0
    foreach i ∈ {2, ..., n} do
        if i is prime then                ▷ 篩で判定
            π +← 1
    return π
```

これは $\Theta(n)$ 時間。

Eratosthenes の篩では $\Theta(n \log(\log(n)))$ 時間だが、用いる篩として線形篩[4]などを採用することで $\Theta(n))$ 時間になる。

[4]詳しくは触れない。今回の話には特に出てこない。

まずは愚直から III

陽に素数を調べる方針では、$O(n^{1-\varepsilon})$ 時間にはできない。

- $[1, n]$ の整数をすべて調べると $\Omega(n)$ 時間かかる。
- $[1, n]$ の素数だけを列挙できたとしても $\Omega(\pi(n))$ 時間かかる[5]。
    - $\pi(n) \sim n/\log(n)$ なので、$\Omega(n/\log(n))$ 時間。

そこで、陽には調べない方針を考える必要がある。

[5] $\pi(n)$ は n 以下の素数の個数。$\pi(1) = 0, \pi(7) = \pi(8) = 4, \pi(12.3) = 5$ など。

篩の復習

Eratosthenes の篩の動作の様子を眺める。

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Figure: 篩の初期状態。

篩の復習

Eratosthenes の篩の動作の様子を眺める。

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Figure: 篩。2 で篩っている様子。

篩の復習

Eratosthenes の篩の動作の様子を眺める。

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Figure: 篩。3 で篩っている様子。

篩の復習

Eratosthenes の篩の動作の様子を眺める。

	2	3	~~4~~	5	~~6~~	7	~~8~~	~~9~~	~~10~~
11	~~12~~	13	~~14~~	~~15~~	~~16~~	17	~~18~~	19	~~20~~
~~21~~	~~22~~	23	~~24~~	25	~~26~~	~~27~~	~~28~~	29	~~30~~
31	~~32~~	~~33~~	~~34~~	35	~~36~~	37	~~38~~	~~39~~	~~40~~
41	~~42~~	43	~~44~~	~~45~~	~~46~~	47	~~48~~	49	~~50~~
~~51~~	~~52~~	53	~~54~~	55	~~56~~	~~57~~	~~58~~	59	~~60~~
61	~~62~~	~~63~~	~~64~~	65	~~66~~	67	~~68~~	~~69~~	~~70~~
71	~~72~~	73	~~74~~	~~75~~	~~76~~	77	~~78~~	79	~~80~~
~~81~~	~~82~~	83	~~84~~	85	~~86~~	~~87~~	~~88~~	89	~~90~~
91	~~92~~	~~93~~	~~94~~	95	~~96~~	97	~~98~~	~~99~~	~~100~~

Figure: 篩。素数でないとわかった 4 では篩わない。

篩の復習

Eratosthenes の篩の動作の様子を眺める。

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Figure: 篩。5 で篩っている様子。

篩の復習

Eratosthenes の篩の動作の様子を眺める。

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Figure: 篩。素数でないとわかった 6 では篩わない。

篩の復習

Eratosthenes の篩の動作の様子を眺める。

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Figure: 篩。7 で篩っている様子。

篩の復習

Eratosthenes の篩の動作の様子を眺める。

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Figure: 篩。素数でないとわかった 8 から 10 では篩わない。

篩の復習

Eratosthenes の篩の動作の様子を眺める。

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Figure: 篩。$\sqrt{100}$ 以下の素数で篩った様子。残りは素数。

重要な観察

「素数 i によって篩われる整数は何個あるか？」

- これを $2 \leqslant i \leqslant \sqrt{n}$ の各素数について考える。
- それらの和を $n-1$ から引けば n 以下の素数の個数がわかる。

→ というわけで、これを高速に求めたい。

篩われる個数を求める I

どのような数が篩われるかを考える。

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Figure: 3 で篩っている様子。

$$\boxed{} = \{9, 15, 21, 27, 33, 39, 45\}$$
$$= \{3 \cdot j \mid j \in \{3, 5, 7, 9, 11, 13, 15\}\}.$$

篩われる個数を求める II

i で篩われる個数は、以下の値の差から求められる。

- $\lfloor n/i \rfloor$ 以下のうち、i 未満の素数では篩われなかった個数
  - $(i, n) = (3, 50)$ では $|\{2, 3, 5, 7, 9, 11, 13, 15\}| = 8$
- i 未満の整数の個数
  - $i = 3$ では $|\{2\}| = 1$

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Figure: $\{3 \cdot j \mid j \in \{3, 5, 7, 9, 11, 13, 15\}\}$ の 7 個が 3 で篩われる様子。

篩われる個数を求める III

以下の値は、i 以前に篩われていることに注意。

- i 未満の素数 j に対し、$i \cdot j$。
- i 未満の素数で篩われた整数 j に対し、$i \cdot j$。

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60

Figure: $\{5 \cdot j \mid j \in \{5, 7, 11\}\}$ が 5 で篩われる様子。

たとえば $15 = 5 \cdot 3$ や、$20 = 5 \cdot (2 \cdot 2)$ は、すでに篩われている。

篩われる個数を求める IV

以下の値を求めればよいとわかった。

- $\lfloor n/i \rfloor$ 以下のうち、i 未満の素数では篩われなかった個数
- i 未満の整数の個数 $\pi(i-1)$

そこで、以下のようにおく。

$$S_i(v) := v \text{ 以下のうち } i \text{ 以下の素数では篩われなかった個数}$$

i で篩うとき、得ているのは $S_{i-1}(v)$ で、得たいのは $S_i(v)$。
特に、初め $S_1(v) = v - 1$。また、$S_{i-1}(i-1) = \pi(i-1)$。

篩われる個数を求める V

$S_i(v)$ を求めたい。

- i が素数でない場合
    - $S_i(v) = S_{i-1}(v)$
    - 篩う処理をしないため。
- $i^2 > v$ の場合
    - $S_i(v) = S_{i-1}(v)$
    - 判明する最小の合成数 i^2 が範囲外のため。

篩われる個数を求める VI

$S_i(v)$ を求めたい。$i^2 \leqslant v$ なる素数 i について考える。

i で篩われる個数は

- $\lfloor v/i \rfloor$ 以下のうち、i 未満の素数では篩われなかった個数
- i 未満の整数の個数

の差だったので、

$$S_i(v) = S_{i-1}(v) - (S_{i-1}(\lfloor v/i \rfloor) - \pi(i-1))$$

とわかる。

篩われる個数を求める VII

以下を求めたくなった。

$$S_i(n) = S_{i-1}(n) - (S_{i-1}(\lfloor n/i \rfloor) - \pi(i-1)).$$

右辺に $S_{i-1}(\lfloor n/i \rfloor)$ があるため、$S_*(\lfloor n/* \rfloor)$ も求める必要がある[6]。

ここで、$\lfloor \lfloor n/i \rfloor /j \rfloor = \lfloor n/ij \rfloor$ に注意すると、$\lfloor n/* \rfloor$ の取りうる値は

$$\underbrace{1, 2, \ldots, \lfloor \sqrt{n} \rfloor}_{i\ (1 \leqslant i \leqslant \sqrt{n})}, \underbrace{\lfloor n/\lfloor \sqrt{n} \rfloor \rfloor, \ldots, \lfloor n/2 \rfloor, n}_{\lfloor n/i \rfloor\ (1 \leqslant i \leqslant \sqrt{n})}$$

の $O(\sqrt{n})$ 通りしかないことがわかる[7, 8]。

[6] $*$ は wild card。
[7] $\lfloor n/\lfloor \sqrt{n}+1 \rfloor \rfloor \leqslant \lfloor n/\sqrt{n} \rfloor = \lfloor \sqrt{n} \rfloor$ から従う。
[8] $\lfloor \sqrt{n} \rfloor = \lfloor n/\lfloor \sqrt{n} \rfloor \rfloor$ は成り立ったり成り立たなかったりするので注意。

Lucy DP I

よって、長さ $O(\sqrt{n})$ の配列[9]を管理して DP すればよい。

更新順に気をつければ、DP 配列を使い回して

$$\mathrm{dp}[n/j] \xleftarrow{-} \mathrm{dp}[\lfloor n/(i \cdot j) \rfloor] - \pi(i-1)$$

と更新できる。

$n/j \geqslant i^2$ なる (i, j) についてのみ更新するように気をつける。

考案者 Lucy_Hedgehog の名前から、主に Project Euler 界隈では Lucy DP と呼ばれている。

[9] $i, \lfloor n/i \rfloor$ $(1 \leqslant i \leqslant \sqrt{n})$ で 2 本持つなり、配列の前後で分けるなりする。

Lucy DP II — 擬似コード

Algorithm 2.3: Lucy DP

function $\textsc{PrimeCount-Lucy}(n)$

- $R \leftarrow (\lfloor n/i \rfloor - 1)_{i=1}^{\lfloor\sqrt{n}\rfloor}$
- $L \leftarrow (i-1)_{i=1}^{\lfloor n/\lfloor\sqrt{n}\rfloor\rfloor}$
- **foreach** $i \leftarrow (2, 3, \ldots, \lfloor\sqrt{n}\rfloor)$ **do**
  - **if** $L_i \leqslant L_{i-1}$ **then continue** ▷ i is prime $\iff L_i > L_{i-1}$
  - $\pi_{i-1} = L_{i-1}$ ▷ $\pi_{i-1} = \pi(i-1)$
  - **foreach** $j \leftarrow (1, 2, \ldots, \lfloor\sqrt{n}\rfloor)$ **do**
    - **if** $\lfloor n/j \rfloor < i^2$ **then break**
    - $(L_{\lfloor n/j \rfloor}$ **or** $R_j) \overset{-}{\leftarrow} (L_{\lfloor n/ij \rfloor}$ **or** $R_{ij}) - \pi_{i-1}$
  - **foreach** $j \leftarrow (\lfloor n/\lfloor\sqrt{n}\rfloor\rfloor - 1, \ldots, 2, 1)$ **do**
    - **if** $j < i^2$ **then break**
    - $L_j \overset{-}{\leftarrow} L_{\lfloor j/i \rfloor} - \pi_{i-1}$
  - **return** R_1

Lucy DP III

擬似コード中の A_i **or** B_j は

- A_i が定義されていれば A_i
- そうでなければ B_j

を意味するものとする。

$S_{i-1}(i) > S_{i-1}(i-1)$ のとき、i が素数となることに注意せよ[10]。
ループ先頭において、$S_{i-1}(v) = L(v)$ である。

また、10 行目のループの都合で、L の長さを $\lfloor n/\lfloor\sqrt{n}\rfloor\rfloor$ とした。

[10] 個数の差分を見れば、条件を満たすかの判定ができるということ。

Lucy DP IV — 計算量解析

9 行目と 12 行目の実行回数を見積もる。

7 行目より $j \leqslant \lfloor\sqrt{n}\rfloor \leqslant \sqrt{n}$、8 行目より $n/j \geqslant \lfloor n/j\rfloor \geqslant i^2$ で[11]、
9 行目の実行回数は高々 $\min\{\sqrt{n}, n/i^2\}$。

10 行目より $j < \lfloor n/\lfloor\sqrt{n}\rfloor\rfloor = \sqrt{n} + o(1)$、11 行目より $j \geqslant i^2$ で、
12 行目の実行回数は高々 $\max\{\sqrt{n} - i^2 + o(1), 0\}$。

これらを、各素数 i について足し合わせればよい。

[11] j について不等式を解き、j の取る範囲が実行回数に相当する。

Lucy DP V — 計算量解析

$$\begin{aligned}
&\int_{x=2}^{\sqrt{n}} \min\{\sqrt{n}, n/x^2\}\, d\pi(x) + \int_{x=2}^{\sqrt{n}} \max\{\sqrt{n} - x^2, 0\}\, d\pi(x) \\
&= \sqrt{n} \int_2^{\sqrt[4]{n}} d\pi(x) + n \int_{\sqrt[4]{n}}^{\sqrt{n}} \frac{d\pi(x)}{x^2} + \int_2^{\sqrt[4]{n}} (\sqrt{n} - x^2)\, d\pi(x) \\
&= 2\sqrt{n} \int_2^{\sqrt[4]{n}} d\pi(x) + n \int_{\sqrt[4]{n}}^{\sqrt{n}} \frac{d\pi(x)}{x^2} - \int_2^{\sqrt[4]{n}} x^2\, d\pi(x).
\end{aligned}$$

ここで

$$\pi(x) \sim \frac{x}{\log(x)}$$

であり、

$$d\pi(x) \sim \frac{\log(x) - 1}{\log(x)^2}\, dx.$$

Lucy DP VI — 計算量解析

$\mathrm{Ei}(\log(x)) = \mathrm{li}(x) \sim x/\log(x)$ より、

$$
\begin{aligned}
\int \frac{d\pi(x)}{x^2} &\sim \int \frac{\log(x)-1}{x^2\log(x)^2}\,dx \\
&= \frac{1}{x\log(x)} + 2\,\mathrm{Ei}(-\log(x)) + \mathrm{const} \\
&= \frac{1}{x\log(x)} + 2\,\mathrm{li}(x^{-1}) + \mathrm{const} \\
&\sim -\frac{1}{x\log(x)}.
\end{aligned}
$$

ここで、$\mathrm{Ei}(x)$ と $\mathrm{li}(x)$ はそれぞれ指数積分と対数積分を表す[12]。

[12] この辺は Wolfram|Alpha や Integral Calculator を頼った。

Lucy DP VII — 計算量解析

残りの項も同様に計算する。

$$
\begin{aligned}
\int x^2 \, d\pi(x) &\sim \int \frac{x^2(\log(x) - 1)}{\log(x)^2} \, dx \\
&= \frac{x^3}{\log(x)} - 2\,\mathrm{Ei}(3\log(x)) + \mathrm{const} \\
&= \frac{x^3}{\log(x)} - 2\,\mathrm{li}(x^3) + \mathrm{const} \\
&\sim \frac{x^3}{3\log(x)}.
\end{aligned}
$$

Lucy DP VIII — 計算量解析

以上より、

$$
\begin{aligned}
&2\sqrt{n}\int_2^{\sqrt[4]{n}} d\pi(x) + n\int_{\sqrt[4]{n}}^{\sqrt{n}} \frac{d\pi(x)}{x^2} - \int_2^{\sqrt[4]{n}} x^2\,d\pi(x) \\
&\sim 2\sqrt{n}\left[\frac{x}{\log(x)}\right]_2^{\sqrt[4]{n}} - n\left[\frac{1}{x\log(x)}\right]_{\sqrt[4]{n}}^{\sqrt{n}} - \left[\frac{x^3}{3\log(x)}\right]_2^{\sqrt[4]{n}} \\
&= \cdots \\
&= O\left(\frac{n^{3/4}}{\log(n)}\right).
\end{aligned}
$$

実際には、$d\pi(x)$ が絡む積分は、係数を気にしなければ、dx で積分して $\log(n)$ で割ってもうまくいくことが多そう[13]。

[13] ところで **Riemann–Stieltjes 積分**とかで調べるとよい？

Lucy DP IX — 総和への応用

$S_i(v)$ の代わりに以下のようにおく。

$S^1_i(v) :=$ v 以下のうち i 以下の素数で篩われていない素数の総和

初期化と更新は以下の通り。

$$S^1_1(v) = \sum_{i=2}^{v} i = \left\lfloor \frac{v \cdot (v+1)}{2} \right\rfloor - 1,$$
$$S^1_i(v) = S^1_{i-1}(v) - i \cdot (S^1_{i-1}(\lfloor v/i \rfloor) - S^1_{i-1}(i-1)).$$

同様にして、2 乗和 $S^2_i(v)$ なども求められる[14]。

[14] 経緯としては、元々は総和 $S^1_i(v)$ を求める問題の解法として提案された。

乗法的関数について

以下を満たす関数 f を**乗法的関数** (*multiplicative function*) と呼ぶ。

- $f(1) = 1$, and
- $\gcd(u, v) = 1 \implies f(uv) = f(u) \cdot f(v)$.

たとえば、Euler の ϕ 関数は乗法的関数である。特に、

$$\phi\left(\prod_{p:\,\mathrm{prime}} p^{e_p}\right) = \prod_{p:\,\mathrm{prime}} (p-1) \cdot p^{e_p - 1}$$

が成り立つ。

e.g., $\phi(120) = \phi(2^3 \cdot 3 \cdot 5) = \underbrace{\phi(2^3)}_{4} \cdot \underbrace{\phi(3)}_{2} \cdot \underbrace{\phi(5)}_{4} = 32$。

乗法的関数の和

乗法的関数 f に対して、$\sum_{i=1}^{n} f(i)$ を高速に求めたくなる。

例として、1 以上 n 以下の整数の組のうち、互いに素なものは

$$\sum_{i=1}^{n} \phi(i)$$

と表せる（順序は区別しないとする）。

さて、$1 < i \leqslant n$ の親を $i/\mathrm{gpf}(i)$ とする n 頂点の木を考えてみる。
ここで、$\mathrm{gpf}(i)$ は i の最大の素因数 (greatest prime factor) である。

図を次のページに載せる。

木

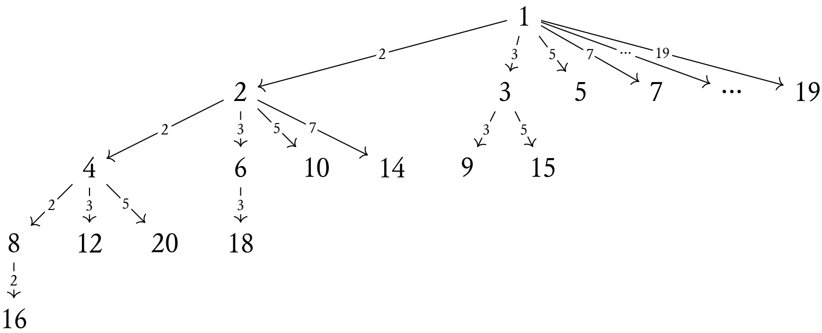


Figure: $n = 20$ の木

子での値の和 I

i の子は、$j \in [\mathrm{gpf}(i), n/i]$ の各素数 j について $i \cdot j$ と表せる。

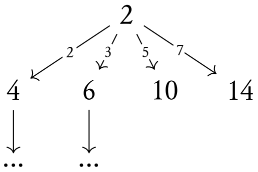


Figure: $n = 20$ の木における $i = 2$ の部分木

$j = \mathrm{gpf}(i)$ とそれ以外の子に分けて、次のように表せる。

$$f(4) + f(6) + f(10) + f(14) = f(2^2) + f(2) \cdot (f(3) + f(5) + f(7)).$$

子での値の和 II

素因数が複数ある場合は少し注意が必要。

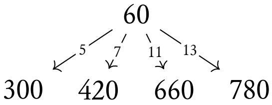


Figure: ある n の木における $i = 60$ の子

$j = \mathrm{gpf}(i)$ とそれ以外の子に分けて、次のように表せる。

$$
\begin{aligned}
&f(300) + f(420) + f(660) + f(780) \\
&= f(12) \cdot f(5^2) + f(60) \cdot (f(7) + f(11) + f(13)).
\end{aligned}
$$

木上の DFS I

葉でない頂点 $v = (\prod_p p^{e_p}) \cdot q^c$ $(\mathrm{gpf}(v) = q)$ にいるとき、

- $f(\prod_p p^{e_p}) \cdot f(q^{c+1})$
- $f(v) \cdot \sum_r f(r)$
  - r は $q < r \leqslant n/v$ を満たす素数

の和を求めればよい。葉でない頂点のみ探索するとする。

$f(\prod_p p^{e_p})$ や $f(v)$、q^c などは DFS しながら管理すればよい[15]。

[15] 最大でない素因数の f と、最大素因数を分けて持てばよい。

木上の DFS II

$f(q^c)$ と $\sum_r f(r)$ を高速に求められる必要がある。

- $f(q^c)$ の計算
  - (q, c, q^c) などから $O(1)$ 時間で求まるのが望ましい。
- $\sum_r f(r)$ の計算・前処理
  - $f(r)$ が多項式なら Lucy DP などで求められる。
  - 高速に求められるなら、多項式でなくてもよい。

木上の DFS III — 計算量解析（未解決）

葉以外の頂点 $\{i \mid i \cdot \mathrm{gpf}(i) \leqslant n\}$ の個数を求めればよい。

$p \leqslant \sqrt[4]{n}$ なる各素数 p について、

- $i \cdot p \leqslant n$, and
- $\mathrm{gpf}(i) = p$

なる i が $O(\sqrt{n})$ 個であれば、$O(n^{3/4} / \log(n))$ 個と示せる。

上記は未解決だが、$n \leqslant 10^{12}$ の範囲では成り立っているそう。

See: https://zhuanlan.zhihu.com/p/33544708.

Lucy DP の高速化

Lucy DP を $O(n^{2/3})$ 時間に高速化する。

元々の計算量は

$$\int_2^{\sqrt{n}} \min\{\sqrt{n}, n/x^2\} d\pi(x) + \int_2^{\sqrt{n}} \max\{\sqrt{n} - x^2, 0\} d\pi(x)$$

に由来するが、区間 $[2, \sqrt[6]{n}]$ と $[\sqrt[3]{n}, \sqrt{n}]$ での積分を考えてみる。

定積分 I

$$
\begin{aligned}
&\int_2^{\sqrt[6]{n}} \min\{\sqrt{n}, n/x^2\}\, d\pi(x) + \int_2^{\sqrt[6]{n}} \max\{\sqrt{n} - x^2, 0\}\, d\pi(x) \\
&= \sqrt{n} \int_2^{\sqrt[6]{n}} d\pi(x) + \int_2^{\sqrt[6]{n}} (\sqrt{n} - x^2)\, d\pi(x) \\
&= 2\sqrt{n} \int_2^{\sqrt[6]{n}} d\pi(x) - \int_2^{\sqrt[6]{n}} x^2\, d\pi(x) \\
&\sim 2\sqrt{n} \left[\frac{x}{\log(x)}\right]_2^{\sqrt[6]{n}} - \left[\frac{x^3}{3\log(x)}\right]_2^{\sqrt[6]{n}} \\
&= O\left(\frac{n^{2/3}}{\log(n)}\right).
\end{aligned}
$$

定積分 II

$$
\begin{aligned}
&\int_{\sqrt[3]{n}}^{\sqrt{n}} \min\{\sqrt{n}, n/x^2\}\, d\pi(x) + \int_{\sqrt[3]{n}}^{\sqrt{n}} \max\{\sqrt{n} - x^2, 0\}\, d\pi(x) \\
&= n \int_{\sqrt[3]{n}}^{\sqrt{n}} \frac{d\pi(x)}{x^2} \\
&\sim n \left[-\frac{1}{x \log(x)} \right]_{\sqrt[3]{n}}^{\sqrt{n}} \\
&= O\left(\frac{n^{2/3}}{\log(n)} \right).
\end{aligned}
$$

場合分け

これにより、$i \in [2, \sqrt[6]{n}] \cup [\sqrt[3]{n}, \sqrt{n}]$ なる素数 i では、そのまま Lucy DP をしても $O(n^{2/3}/\log(n))$ 時間で抑えられるとわかる。

そこで、残りの $(\sqrt[6]{n}, \sqrt[3]{n})$ の区間について考える。

$\mathrm{dp}[n/j]$ の更新に関して、$n/j \geqslant n^{2/3}$ すなわち $j \leqslant \sqrt[3]{n}$ のときは、愚直に更新しても $O(n^{2/3}/\log(n))$ 回で済む[16]。

あとは、$n/j < n^{2/3}$ について考えればよい。

[16] i が高々 $n^{1/3}/\log(n)$ 個、j が高々 $n^{1/3}$ 個なので。

重要な事実

以下の事実に気をつける。

- i で篩われる合成数 v について、$\mathrm{lpf}(v) = i$ が成り立つ。
  - $\mathrm{lpf}(v)$ は v の最小の素因数 (least prime factor) を表す。
  - ここでは $\mathrm{lpf}(1) = \infty$ としておく[17]。
- $\mathrm{lpf}(v) \geqslant i$ なる $v \leqslant n^{2/3}$ は $\Theta(n^{2/3}/\log(n))$ 個。
  - 解析に関しては後述の関数を参照。

→ $\mathrm{lpf}(v) = i$ なる v を一つあたり $O(1)$ 時間で列挙できれば、$i \in (n^{1/6}, n^{1/3})$ で篩われる数を陽に列挙しても大丈夫。

[17] 有限の整数で 1 を篩うことはできないため？

$\operatorname{lpf}(v) = i$ なる合成数 v の列挙 I

乗法的関数の和を求める際に作ったのと同様の木を DFS する。

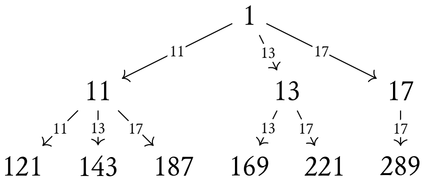


Figure: 木の一部分

根 1 から最初に辿った値が最小素因数となることに注意する。
たとえば、$\operatorname{lpf}(187) = 11$ とわかる。

素数 i の深さは 1 であり、i の真の部分木の各数が求める v である。

$\mathrm{lpf}(\nu) = i$ なる合成数 ν の列挙 II

$n^{2/3}$ 以下の合成数を列挙する際の空間計算量を確認する。

再帰で行う場合、深さは $\log(n)$ 段になるので問題ない。

stack を用いる場合について考える。素数 $i \in (n^{1/6}, n^{1/3})$ の子は

$$\frac{n^{2/3}/\log(n)}{n^{1/6}}$$

個程度あり、深さは高々 $\log(n)$ 段なので、$O(\sqrt{n})$ space で済む[18]。

[18] 深くなるにつれて子は減るので、粗い見積もりではありそう。

高速化の方針

v が篩われていれば $b_v = 1$、そうでないとき $b_v = 0$ となる配列を

- v を篩うとき、$b_v \xleftarrow{+} 1$ で更新する。
- v 以下の篩われた個数は、$\sum_{i=1}^{v} b_i$ で取得する。

と管理すればよい。→ BIT を用いて $O(\log(n))$ 時間で可能[19]。

$n/j \geqslant n^{2/3}$ の Lucy DP と併せて、個数を求めることができる[20]。

更新は $O(n^{2/3}/\log(n))$ 回なので、$O(n^{2/3})$ 時間となる。

[19] 実際には、$v = \lfloor n/* \rfloor$ のみ管理すればよいので、$O(\sqrt{n})$ space にできる。

[20] $n^{2/3}$ 未満の範囲については Lucy DP をする代わりに、$n^{2/3}$ 以上の範囲の DP のための補助情報（篩われた個数）のみを管理するということ。

場合分けの remark

- $2 \leqslant i \leqslant n^{1/6}$
  - そのまま Lucy DP しても $O(n^{2/3}/\log(n))$ 時間。
- $n^{1/6} < i < n^{1/3}$
  - $n/j \geqslant n^{2/3}$
    - Lucy DP で $O(n^{2/3}/\log(n))$ 回の更新。
    - 更新の際は BIT から値を取得し、$O(n^{2/3})$ 時間。
  - $n/j < n^{2/3}$
    - $\mathrm{lpf}(v) = i$ なる合成数 v を列挙して BIT で管理する。
    - 操作ごとに $O(\log(n))$ 時間なので $O(n^{2/3})$ 時間。
- $n^{1/3} \leqslant i \leqslant n^{1/2}$
  - そのまま Lucy DP しても $O(n^{2/3}/\log(n))$ 時間。

実装

やや長くなるため、擬似コードは付録に載せる。

ここでは、i と $\lfloor n/i \rfloor$ に対応する配列を分けて持つ方針ではなく、前半が $\lfloor n/i \rfloor$、後半が i に対応する降順の列

$$A = (\perp, n, \lfloor n/2 \rfloor, \ldots, \lfloor n/\lfloor \sqrt{n} \rfloor \rfloor, \lfloor n/\lfloor \sqrt{n} \rfloor \rfloor - 1, \ldots, 2, 1)$$

に対し、$\mathrm{dp}[i]$ を $S_*(A_i)$ に対応させる方針を採用した[21]。

$\lfloor n/k \rfloor$ に対応する要素の添字は

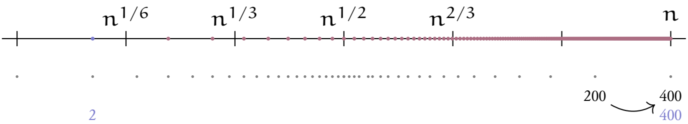
$$\textbf{if } k \leqslant \sqrt{n} \textbf{ then } k \textbf{ else } |A| - \lfloor n/k \rfloor$$

で取得できる。

[21] $\perp$ はダミーの値。

イメージ図

BIT などで管理するイメージ図などを載せてみる。



$$\underbrace{\mathrm{dp}[1]}_{\substack{S_1(400)\\=399}} \xleftarrow{-} \underbrace{\mathrm{dp}[2]}_{\substack{S_1(200)\\=199}} - \underbrace{\pi(2-1)}_{0}$$

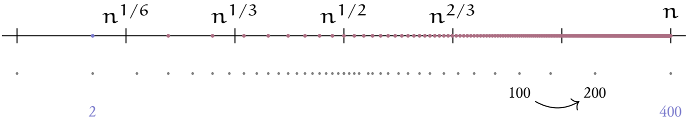
$$\mathrm{dp}[1] = S_2(400) = 200$$

Figure: アルゴリズムの動きの概略[22]

[22] 数直線が対数軸であることに注意。なお、ノーカット版は付録に掲載。

イメージ図

BIT などで管理するイメージ図などを載せてみる。



$$\underbrace{\mathrm{dp}[2]}_{\substack{S_1(200)\\=199}} \xleftarrow{-} \underbrace{\mathrm{dp}[4]}_{\substack{S_1(100)\\=99}} - \underbrace{\pi(2-1)}_{0}$$

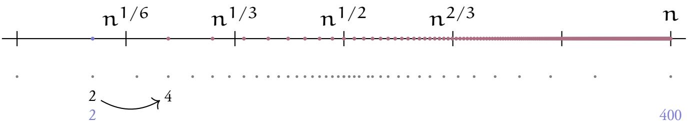
$$\mathrm{dp}[2] = S_2(200) = 100$$

Figure: アルゴリズムの動きの概略[22]

[22]数直線が対数軸であることに注意。なお、ノーカット版は付録に掲載。

イメージ図

BIT などで管理するイメージ図などを載せてみる。



$$\underbrace{\mathrm{dp}[36]}_{\substack{S_1(4)\\=3}} \xleftarrow{-} \underbrace{\mathrm{dp}[38]}_{\substack{S_1(2)\\=1}} - \underbrace{\pi(2-1)}_{0}$$

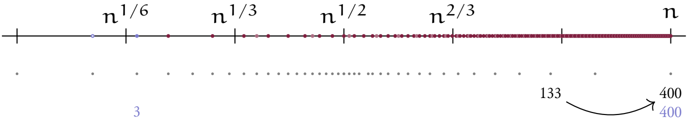
$$\mathrm{dp}[36] = S_2(4) = 2$$

Figure: アルゴリズムの動きの概略[22]

[22] 数直線が対数軸であることに注意。なお、ノーカット版は付録に掲載。

イメージ図

BIT などで管理するイメージ図などを載せてみる。



$$\underbrace{\mathrm{dp}[1]}_{\substack{S_2(400)\\=200}} \overset{-}{\longleftarrow} \underbrace{\mathrm{dp}[3]}_{\substack{S_2(133)\\=67}} - \underbrace{\pi(3-1)}_{1}$$

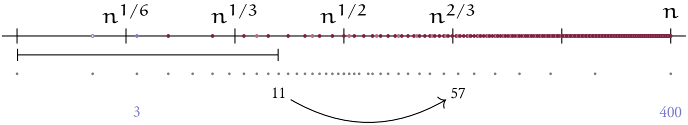
$$\mathrm{dp}[1] = S_3(400) = 134$$

Figure: アルゴリズムの動きの概略[22]

[22] 数直線が対数軸であることに注意。なお、ノーカット版は付録に掲載。

イメージ図

BIT などで管理するイメージ図などを載せてみる。



$$\underbrace{\mathrm{dp}[7]}_{\substack{S_3(57)\\=20}} \overset{-}{\longleftarrow} \underbrace{\mathrm{dp}[29]}_{\substack{S_2(11)\\=6}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 11,\ \sqrt[6]{n}<\mathrm{lpf}(v)<5\right\}\right|}_{\substack{|\{9\}|\\=1}} - \underbrace{\pi(5-1)}_{2}$$

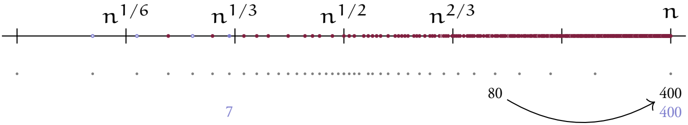
$$\mathrm{dp}[7] = S_5(57) = 17$$

Figure: アルゴリズムの動きの概略[22]

[22] 数直線が対数軸であることに注意。なお、ノーカット版は付録に掲載。

イメージ図

BIT などで管理するイメージ図などを載せてみる。



$$\underbrace{\mathrm{dp}[1]}_{\substack{S_3(400)\\=134}} \xleftarrow{-} \underbrace{\mathrm{dp}[5]}_{\substack{S_3(80)\\=28}} - \underbrace{\pi(5-1)}_{2}$$

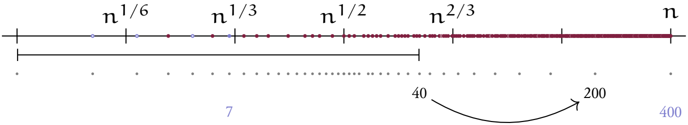
$$\mathrm{dp}[1] = S_5(400) = 108$$

Figure: アルゴリズムの動きの概略[22]

[22] 数直線が対数軸であることに注意。なお、ノーカット版は付録に掲載。

イメージ図

BIT などで管理するイメージ図などを載せてみる。



$$\underbrace{\mathrm{dp}[2]}_{\substack{S_3(200)\\=68}} \longleftarrow \underbrace{\mathrm{dp}[10]}_{\substack{S_2(40)\\=20}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 40,\ \sqrt[6]{n}<\mathrm{lpf}(v)<5\right\}\right|}_{\substack{|\{9,15,21,27,33,39\}|\\=6}} - \underbrace{\pi(5-1)}_{2}$$

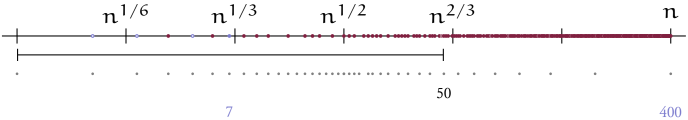
$$\mathrm{dp}[2] = S_5(200) = 56$$

Figure: アルゴリズムの動きの概略[22]

[22] 数直線が対数軸であることに注意。なお、ノーカット版は付録に掲載。

イメージ図

BIT などで管理するイメージ図などを載せてみる。



$$\underbrace{\mathrm{dp}[8]}_{\substack{S_2(50)\\=25}} \overset{-}{\longleftarrow} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 50,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15,21,25,27,33,35,39,45\}|\\=9}}$$

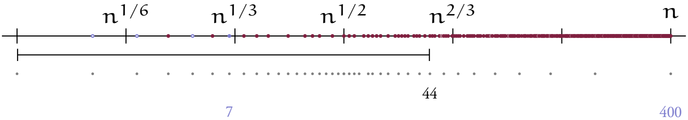
$$\mathrm{dp}[8] = S_5(50) = 16$$

Figure: アルゴリズムの動きの概略[22]

[22] 数直線が対数軸であることに注意。なお、ノーカット版は付録に掲載。

イメージ図

BIT などで管理するイメージ図などを載せてみる。



$$\underbrace{\mathrm{dp}[9]}_{\substack{S_2(44)\\=22}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 44,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15,21,25,27,33,35,39\}|\\=8}}$$

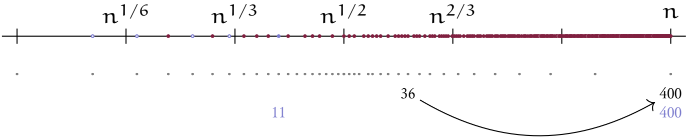
$$\mathrm{dp}[9] = S_5(44) = 14$$

Figure: アルゴリズムの動きの概略[22]

[22] 数直線が対数軸であることに注意。なお、ノーカット版は付録に掲載。

イメージ図

BIT などで管理するイメージ図などを載せてみる。



$$\underbrace{dp[1]}_{\substack{S_7(400)\\=94}} \overset{-}{\longleftarrow} \underbrace{dp[11]}_{\substack{S_7(36)\\=11}} - \underbrace{\pi(11-1)}_{4}$$

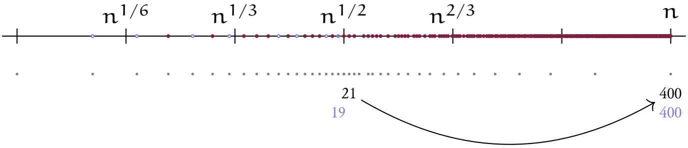
$$dp[1] = S_{11}(400) = 87$$

Figure: アルゴリズムの動きの概略[22]

[22] 数直線が対数軸であることに注意。なお、ノーカット版は付録に掲載。

イメージ図

BIT などで管理するイメージ図などを載せてみる。



$$\underbrace{\mathrm{dp}[1]}_{\substack{S_{17}(400)\\=79}} \overset{-}{\longleftarrow} \underbrace{\mathrm{dp}[19]}_{\substack{S_{17}(21)\\=8}} - \underbrace{\pi(19-1)}_{7}$$

$$\mathrm{dp}[1] = S_{19}(400) = 78$$

Figure: アルゴリズムの動きの概略[22]

[22]数直線が対数軸であることに注意。なお、ノーカット版は付録に掲載。

解析に関する関数たち

- $\Phi(x, y)$
  - x 以下の正整数のうち、y-rough である個数を表す。
    - y-rough：最小素因数が y 以上
  - $\Phi(x, x^{1/u}) \sim x \cdot \omega(u) / \log(x^{1/u})$
    - $\omega(u)$ は Buchstab function と呼ばれる。
- $\Psi(x, y)$
  - x 以下の正整数のうち、y-smooth である個数を表す。
    - y-smooth (y-friable)：最大素因数が y 以下
  - $\Psi(x, x^{1/a}) \sim x \cdot \rho(a)$
    - $\rho(a)$ は Dickman–de Bruijn function と呼ばれる。
    - $\rho(a) \approx a^{-a}$。

さらなる高速化

区間の分け方を調整することで、log factor を減らせる。

- $2 \leqslant i \leqslant n^{1/6}$
- $n^{1/6} < i < n^{1/3} / \log(n)^{2/3}$
    - $n/j < n^{2/3} / \log(n)^{1/3}$ で分ける。
- $n^{1/3} / \log(n)^{2/3} \leqslant i \leqslant n^{1/2}$

各分岐で行うことは同じ。$O(n^{2/3} / \log(n)^{1/3})$ 時間になる。

高速化の概略

乗法的関数の和の計算を高速化する。

以下の流れで求める。

1. 素数 p に対して $f(p)$ の和を求める。
2. $(\sqrt[3]{n}+1)$-rough number i に対して $f(i)$ の和を求める。
3. $(\sqrt[6]{n}+1)$-rough number i に対して $f(i)$ の和を求める。
4. 整数 i に対して $f(i)$ の和を求める。

具体的な p や i の範囲などは次ページ以降で説明する。

記法の導入

- p_k：k 番目の素数 (e.g. $p_1 = 2, p_2 = 3, p_3 = 5, \ldots$)
    - $(p_{\pi(i)})_{i=2}^{\infty} = (2, 3, 3, 5, 5, 7, 7, 7, 7, 11, 11, \ldots)$ に注意。
- $S_{\mathbb{P}}^f(n) := \sum_{\substack{2 \leqslant p \leqslant n \\ p\colon \text{prime}}} f(p)$：素数における f の和
- $S_k^f(n) := \sum_{\substack{1 \leqslant i \leqslant n \\ \mathrm{lpf}(i) \geqslant p_k}} f(i)$：$p_k$-rough number における f の和
- $L(f, n) := (f(1), f(2), \ldots, f(\lfloor \sqrt{n} \rfloor))$
- $R(f, n) := (f(\lfloor n/\lfloor \sqrt{n} \rfloor \rfloor), \ldots, f(\lfloor n/2 \rfloor), f(n))$
- $V(f, n) := (L(f, n), R(f, n))$

素数での f の和

まず、$V(S^f_{\mathbb{P}}, n)$ を計算する。

Remark:

$$S^f_{\mathbb{P}}(n) := \sum_{\substack{2 \leqslant p \leqslant n \\ p:\ \text{prime}}} f(p).$$

各 $v = i$ $(1 \leqslant i \leqslant \lfloor\sqrt{n}\rfloor)$ と $v = \lfloor n/i \rfloor$ $(1 \leqslant i \leqslant \lfloor\sqrt{n}\rfloor)$ に対して、v 以下の素数 p における $f(p)$ の総和を求めるということ。

これは、素数 p に対して $f(p) = g(p)$ なる多項式 g が存在すれば、先の高速化した Lucy DP で $O(n^{2/3})$ 時間で計算できる[23]。

[23] 素数 p 以外の部分は無視して、素数の部分さえ多項式で表せればよい。

$(\sqrt[3]{n}+1)$-rough number での f の和 I

$(\sqrt[3]{n}+1)$-rough number での f の和 $V(S^f_{\pi(\sqrt[3]{n})+1}, n)$ を求める。

$\lfloor n/i \rfloor$ の値の範囲によって分けて考える。なお、簡便さのため、$\sqrt[3]{n}$ を超える最小の素数 $p_{\pi(\sqrt[3]{n})+1}$ を q とおく。

まず、$q > \sqrt[3]{n}$ から、$n^{1/3}$ 以下の q-rough number は 1 のみ。

$m = \lfloor n/i \rfloor \leqslant n^{1/3}$ について、

$$S^f_{\pi(\sqrt[3]{n})+1}(\lfloor n/i \rfloor) = f(1)$$

より、各 m について O(1) 時間で計算できる。

$(\sqrt[3]{n}+1)$-rough number での f の和 II

$q > \sqrt[3]{n}$ から、$n^{2/3}$ 以下の q-rough number の素因数は高々 1 つ[24]。

$m = \lfloor n/i \rfloor \in (n^{1/3}, n^{2/3}]$ について、

$$S^f_{\pi(\sqrt[3]{n})+1}(m) = f(1) + (S^f_{\mathbb{P}}(m) - S^f_{\mathbb{P}}(q-1))$$

より[25]、各 m について $O(1)$ 時間で計算できる。

---

[24] $q^2 > n^{2/3}$ なので。
[25] 累積和の差分を求めているだけ。

$(\sqrt[3]{n}+1)$-rough number での f の和 III

$q > \sqrt[3]{n}$ から、n 以下の q-rough number の素因数は高々 2 つ。

$m = \lfloor n/i \rfloor \in (n^{2/3}, n]$ について、

$$S^f_{\pi(\sqrt[3]{n})+1}(m) = f(1) + (S^f_{\mathbb{P}}(m) - S^f_{\mathbb{P}}(q-1)) + \sum_{j=\pi(\sqrt[3]{n})+1}^{\pi(\sqrt{m})} \left(f(p_j^2) + f(p_j)\cdot(S^f_{\mathbb{P}}(m/p_j) - S^f_{\mathbb{P}}(p_j))\right).$$

より[26]、各 m について $O(\pi(\sqrt{m}))$ 時間で計算できる。

[26] 素因数に p_j を 2 つ持つ場合と、p_j と p_j 以外を持つ場合で分ける。

$(\sqrt[3]{n}+1)$-rough number での f の和 IV — 計算量解析

各 $m = \lfloor n/i \rfloor$ について $\Theta(\pi(m))$ 時間かかるので、

$$\sum_{i=1}^{\lfloor n^{1/3} \rfloor} \pi(\sqrt{n/i}) \sim \sum_{i=1}^{\lfloor n^{1/3} \rfloor} \sqrt{n/i} \,/ \log(\sqrt{n/i})$$

$$\sim 2\sqrt{n} \int_1^{n^{1/3}} \frac{dx}{\sqrt{x}\log(n/x)}.$$

ここで、

$$\int \frac{dx}{\sqrt{x}\log(n/x)} = -2\sqrt{n}\ \mathrm{li}(\sqrt{x/n}) + \mathrm{const}$$

$$\sim \frac{4\sqrt{x}}{\log(n/x)}.$$

$(\sqrt[3]{n}+1)$-rough number での f の和 V — 計算量解析

よって、

$$\sum_{i=1}^{n^{1/3}} \pi(\sqrt{n/i}) \sim 8\sqrt{n}\left[\frac{\sqrt{x}}{\log(n/x)}\right]_1^{n^{1/3}} = \Theta(n^{2/3}/\log(n))$$

とわかる。

$(\sqrt[6]{n}+1)$-rough number での f の和 I

$(\sqrt[6]{n}+1)$-rough number での f の和 $V(S^f_{\pi(\sqrt[6]{n})+1}, n)$ を求める。

S^f_{k+1} から S^f_k を計算するための式

$$S^f_k(m) = \sum_{e=0}^{\lfloor \log_{p_k}(m) \rfloor} f(p_k^e) \cdot S^f_{k+1}(\lfloor m/p_k^e \rfloor)$$

を念頭におく[27]。

各 $k = \pi(\lfloor \sqrt[3]{n} \rfloor), \ldots, \pi(\lfloor \sqrt[6]{n} \rfloor) + 1$ について、この式の通り愚直に更新すると $\Theta(n^{5/6}/\log(n))$ 時間かかるため、工夫が必要になる。

[27] p_k の次数ごとに求めて足せばよいということ。

$(\sqrt[6]{n}+1)$-rough number での f の和 II

高速化の方針は Lucy DP のときとほぼ同じ。

$m \geqslant n^{2/3}$ については

$$S_k^f(m) = \sum_{e=0}^{\lfloor \log_{p_k}(m) \rfloor} f(p_k^e) \cdot S_{k+1}^f(\lfloor m/p_k^e \rfloor)$$

で更新し、$m < n^{2/3}$ については差分を BIT で管理する。

$\sqrt{n}$ 以下の素数のみから $n^{2/3}$ 以下の合成数を網羅する部分で、$(\sqrt[6]{n}+1)$-rough であることが効いている。

$(\sqrt[6]{n}+1)$-rough number での f の和 III — 計算量解析

$m \geqslant n^{2/3}$ における更新回数は

$$
\begin{aligned}
\sum_{i=\pi(\sqrt[6]{n})+1}^{\pi(\sqrt[3]{n})} \sum_{j=1}^{\lfloor \sqrt[3]{n} \rfloor} \log_{p_i}(n/j) &\leqslant \sum_{i=\pi(\sqrt[6]{n})+1}^{\pi(\sqrt[3]{n})} \sum_{j=1}^{\lfloor \sqrt[3]{n} \rfloor} \log_{\sqrt[6]{n}}(n) \\
&= \sum_{i=\pi(\sqrt[6]{n})+1}^{\pi(\sqrt[3]{n})} \sum_{j=1}^{\lfloor \sqrt[3]{n} \rfloor} 6 \\
&\sim \frac{n^{1/3}}{\frac{1}{3}\log(n)} \cdot 6 \cdot \lfloor \sqrt[3]{n} \rfloor \\
&= O(n^{2/3}/\log(n))
\end{aligned}
$$

となる。

$(\sqrt[6]{n}+1)$-rough number での f の和 IV — 計算量解析

$m < n^{2/3}$ における更新回数に関する解析は Lucy DP と同じ。

$n^{2/3}$ 以下の $(\sqrt[6]{n}+1)$-rough number の個数に相当し、これは $\Theta(n^{2/3}/\log(n))$ 個であることが知られている。

各操作は BIT で行うため一回あたり $O(\log(n))$ 時間なので、$(\sqrt[6]{n}+1)$-rough number での和は $O(n^{2/3})$ 時間で求められる。

2-rough number での f の和 I

2-rough number での和 $V(S_1^f, n)$、すなわち全体の和を求める[28]。

S_{k+1}^f から S_k^f を計算するための式

$$S_k^f(m) = \sum_{e=0}^{\lfloor \log_{p_k}(m) \rfloor} f(p_k^e) \cdot S_{k+1}^f(\lfloor m/p_k^e \rfloor)$$

を用いて、$k = \pi(\lfloor \sqrt[6]{n} \rfloor), \ldots, 2, 1$ と愚直に更新すればよい。

[28] $\mathrm{lpf}(1) = \infty$ と定義したため、1 も 2-rough であることに注意せよ。

2-rough number での f の和 II — 計算量解析

計算量は以下のようになる。

$$
\begin{aligned}
&\sum_{i=1}^{\pi(\sqrt[6]{n})} \sum_{j=1}^{\lfloor\sqrt{n}\rfloor} (\log_{p_k}(n/i) + \log_{p_k}(i)) \\
&= \sum_{i=1}^{\pi(\sqrt[6]{n})} \sum_{j=1}^{\lfloor\sqrt{n}\rfloor} \log_{p_k}(n) \\
&\leqslant \sum_{i=1}^{\pi(\sqrt[6]{n})} \sum_{j=1}^{\lfloor\sqrt{n}\rfloor} \log_2(n) \\
&\sim \frac{n^{1/6}}{\frac{1}{6}\log(n)} \sqrt{n} \cdot \log_2(n) = O(n^{2/3}).
\end{aligned}
$$

乗法的関数の和の高速化

乗法的関数 f に対する $\sum_{i=1}^{n} f(i)$ が $O(n^{2/3})$ 時間で得られた[29]。

素因数が高々 2 個であるための条件などと絡むため、Lucy DP のときのような方針では、log factor を減らせないと思われる。

なお、未調査だが、$O(n^{2/3}/\log(n))$ time, $O(\sqrt{n})$ space の手法も知られているらしい。

[29] 実際には Θ になるはず。

補足

前述の各手法で p_i を使っているが、必要になるのは $p_i \leqslant \sqrt{n}$ の範囲のみ。予め篩などで列挙しておけばよく、たとえば線形篩を用いれば $\langle O(\sqrt{n}), O(1) \rangle$ time, $O(\sqrt{n})$ space で済む。

単純な乗法的関数として $f(i) = i$ や $f(i) = i^2$ などが挙げられる。テストを行う際にはそれを用いるのが便利だと思われる。

環境

実験に用いた環境は以下の通り。

PC	MacBook Pro (13-inch, M1, 2020)
メモリ	16 GB
言語	Rust, rustc 1.63.0-nightly (`fdca237d5` 2022-06-24)
最適化	`-C opt-level=3`
ツール	Criterion.rs (0.3.5)

計測結果 I

Lucy DP の亜種たちの実測の結果は以下の通り。

いずれもほぼ同じだが、$n = 10^{10}$ 程度で $\Theta(n^{2/3}/\log(n)^{1/3})$ が優勢になっていた。$n = 10^{13}$ で 1.6 秒程度であった。

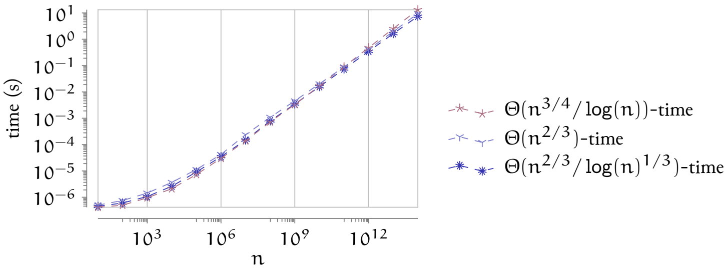


Figure: Lucy DP の計測結果

計測結果 II

Lucy DP の亜種たちの実測の結果は以下の通り。

いずれもほぼ同じだが、$n = 10^{10}$ 程度で $\Theta(n^{2/3}/\log(n)^{1/3})$ が優勢になっていた。$n = 10^{13}$ で 1.6 秒程度であった。

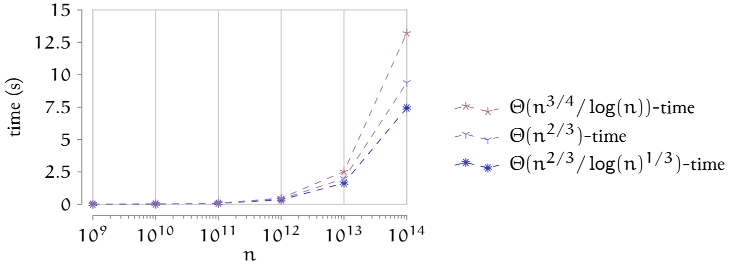


Figure: Lucy DP の計測結果

計測結果 III

乗法的関数の和を求めるアルゴリズムの実測の結果は以下の通り。

こちらも同じような感じ。定数倍が重めだが、$n = 10^9$ 程度で $\Theta(n^{2/3})$ の方が優勢になり始めた。$n = 10^{12}$ で 1.3 秒程度。

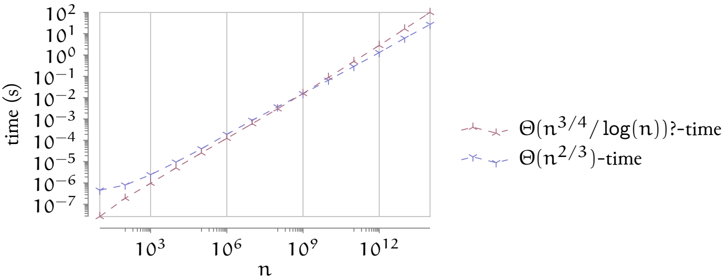


Figure: 乗法的関数の和の計測結果

計測結果 IV

乗法的関数の和を求めるアルゴリズムの実測の結果は以下の通り。

こちらも同じような感じ。定数倍が重めだが、$n = 10^9$ 程度で $\Theta(n^{2/3})$ の方が優勢になり始めた。$n = 10^{12}$ で 1.3 秒程度。

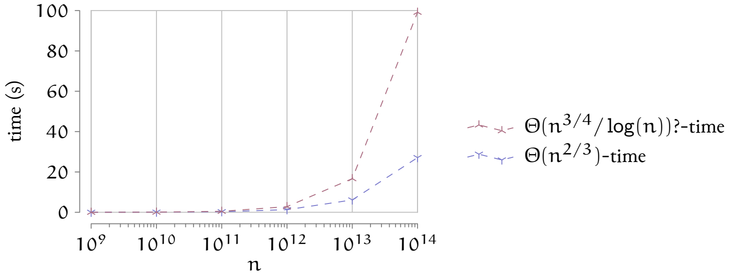


Figure: 乗法的関数の和の計測結果

求めた値

Table: 素数の個数と $\phi(n)$ の和の値

$\log_{10}(n)$	$\pi(n)$	$\sum_{i=1}^{n} \phi(i)$
1	4	32
2	25	3044
3	168	304192
4	1229	30397486
5	9592	3039650754
6	78498	303963552392
7	664579	30396356427242
8	5761455	3039635516365908
9	50847534	303963551173008414
10	455052511	30396355092886216366
11	4118054813	3039635509283386211140
12	37607912018	303963550927059804025910
13	346065536839	30396355092702898919527444
14	3204941750802	3039635509270144893910357854

参考文献 I

- Sum of Multiplicative Function / min-25
  - https://min-25.hatenablog.com/entry/2018/11/11/172216 (web.archive.org:20211009144526)
- min-25 sieve
  - https://zhuanlan.zhihu.com/p/60378354
  - https://oi-wiki.org/math/number-theory/min-25/
- 洲閣篩 (Zhouge sieve)
  - http://debug18.com/posts/calculate-the-sum-of-multiplicative-function (web.archive.org:20190114044154)

参考文献 II

- the black algorithm / baihacker
  - http://baihacker.github.io/main/2020/The_prefix-sum_of_multiplicative_function_the_black_algorithm.html
- Nyaan's Library
  - https://nyaannyaan.github.io/library/multiplicative-function/sum-of-multiplicative-function.hpp
  - https://nyaannyaan.github.io/library/multiplicative-function/prime-counting.hpp
  - https://nyaannyaan.github.io/library/multiplicative-function/prime-counting-o2d3.hpp
  - https://nyaannyaan.github.io/library/multiplicative-function/prime-counting-faster.hpp

関連資料

- 乗法的関数の和を $O(n^{2/3}/\log(n))$ time, $O(\sqrt{n})$ space らしい
  - https://blog.csdn.net/whzzt/article/details/104105025 (web.archive.org:20211009144526)
- 別の手法で $\sum_i \phi(i)$ などを $O(n^{2/3})$ time / maspy
  - https://maspypy.com/dirichlet-積と、数論関数の累積和
- $\pi(n)$: the Meissel, Lehmer, Lagarias, Miller, Odlyzko method
  - $O(x^{2/3}/\log(x)^2)$ time, $O(x^{1/3}\log(x)^3\log(\log(x)))$ space
  - https://www.ams.org/journals/mcom/1996-65-213/S0025-5718-96-00674-6/S0025-5718-96-00674-6.pdf
- 実用的に高速なライブラリ。10^{31} くらいまでできるらしい。
  - https://github.com/kimwalisch/primecount

Thank you!

擬似コード I

$\Theta(n^{2/3})$ 時間の Lucy DP の擬似コード。

拡大して見ればよいので、めちゃくちゃに縮小してある。

Algorithm 8.1: 高速化した Lucy DP

```
function primeCount-lucy^{2/3}(n)
    A = (n, ⌊n/2⌋, ..., ⌊n/⌊\sqrt{n}⌋⌋, ⌊n/⌊\sqrt{n}⌋⌋ - 1, ..., 2, 1)
    S ← (A_1 - 1, A_2 - 1, ..., A_{|A|} - 1)
    π ← 1
    while p_π ⩽ \sqrt[6]{n} do
        foreach i ← (1, 2, ..., |A|) do
            if A_i < p_π^2 then break
            j = (if i · p_π ⩽ \sqrt{n} then i · p_π else |A| - ⌊A_i/p_π⌋)
            S_i \xleftarrow{-} S_j
        π \xleftarrow{+} 1
    b ← (0)_{i=⌊\sqrt[3]{n}⌋}^{|A|}                      ▷ b は BIT で管理する。
    while p_π ⩽ \sqrt[3]{n} do
        foreach i ← (1, 2, ..., ⌊\sqrt[3]{n}⌋) do
            j = (if i · p_π ⩽ \sqrt{n} then i · p_π else |A| - ⌊A_i/p_π⌋)
            if j > \sqrt[3]{n} then
                S_i \xleftarrow{-} (S_j - \sum_{k=j}^{|A|} b_k) - π
            else
                S_i \xleftarrow{-} S_j - π
        foreach v ∈ {v | lpf(v) = p_π, v < n/⌊\sqrt[3]{n}⌋} \ ℙ do
            j = (if v ⩽ \sqrt{n} then |A| - v else ⌊n/v⌋)
            if j > \sqrt[3]{n} then b_j \xleftarrow{+} 1
        π \xleftarrow{+} 1
    foreach j ← (|A|, ..., ⌊\sqrt[3]{n}⌋) do       ▷ ループ範囲に関して^a
        S_j \xleftarrow{-} \sum_{k=j}^{|A|} b_k
    while p_π ⩽ \sqrt{n} do
        foreach i ← (1, 2, ..., |A|) do
            if A_i < p_π^2 then break
            j = (if i · p_π ⩽ \sqrt{n} then i · p_π else |A| - ⌊A_i/p_π⌋)
            S_i \xleftarrow{-} S_j
        π \xleftarrow{+} 1
    return S_1                                      ▷ π(n)
```

[a] $n^{1/6}$ 以下の素数では篩っていることから、BIT で管理されている合成数は $n^{1/3}$ より大きい。そのため $j \leftarrow (|A| - \lfloor\sqrt[3]{n}\rfloor, \ldots, \lfloor\sqrt[3]{n}\rfloor)$ で十分そう。

擬似コード II

$\Theta(n^{2/3}/\log(n)^{1/3})$ 時間の Lucy DP の擬似コード。

拡大して見ればよいので、めちゃくちゃに縮小してある。

Algorithm 8.2: 高速化した Lucy DP

```
function PRIMECOUNT-LUCY^{2/3/LOG}(n)
    A = (n, ⌊n/2⌋, …, ⌊n/⌊\sqrt{n}⌋⌋, ⌊n/⌊\sqrt{n}⌋⌋ − 1, …, 2, 1)
    S ← (A_1 − 1, A_2 − 1, …, A_{|A|} − 1)
    π ← 1
    while p_π ⩽ \sqrt[6]{n} do
        foreach i ← (1, 2, …, |A|) do
            if A_i < p_π^2 then break
            j = (if i · p_π ⩽ \sqrt{n} then i · p_π else |A| − ⌊A_i/p_π⌋)
            S_i ←^{−} S_j
        π ←^{+} 1
    b ← (0)_{i=⌊\sqrt[3]{n}·log(n)^{1/3}⌋}^{|A|}          ▷ b は BIT で管理する。
    if ⌊\sqrt[3]{n} · log(n)^{1/3}⌋ ⩽ \sqrt{n} then
        j_θ = ⌊\sqrt[3]{n} · log(n)^{1/3}⌋
    else
        j_θ = |A| − ⌊n/⌊\sqrt[3]{n} · log(n)⌋⌋
    while p_π ⩽ \sqrt[3]{n} / log(n)^{2/3} do
        foreach i ← (1, 2, …, ⌊\sqrt[3]{n} · log(n)^{1/3}⌋) do
            j = (if i · p_π ⩽ \sqrt{n} then i · p_π else |A| − ⌊A_i/p_π⌋)
            if j > j_θ then
                S_i ←^{−} (S_j − \sum_{k=j}^{|A|} b_k) − π
            else
                S_i ←^{−} S_j − π
        foreach
          v ∈ {v | lpf(v) = p_π, v < n/⌊\sqrt[3]{n} · log(n)^{1/3}⌋} \ ℙ do
            j = (if v ⩽ \sqrt{n} then |A| − v else ⌊n/v⌋)
            if j > j_θ then b_j ←^{+} 1
        π ←^{+} 1
    foreach j ← (|A|, …, ⌊\sqrt[3]{n} · log(n)^{1/3}⌋) do
        S_j ←^{−} \sum_{k=j}^{|A|} b_k
    while p_π ⩽ \sqrt{n} do
        foreach i ← (1, 2, …, |A|) do
            if A_i < p_π^2 then break
            j = (if i · p_π ⩽ \sqrt{n} then i · p_π else |A| − ⌊A_i/p_π⌋)
            S_i ←^{−} S_j
        π ←^{+} 1
    return S_1                                        ▷ π(n)
```

擬似コード III

$\Theta(n^{2/3})$ 時間で乗法的関数の和を求める擬似コード。

拡大して見ればよいので、めちゃくちゃに縮小してある。

Algorithm 8.3: 高速化した乗法的関数の和

```
function MULTIPLICATIVEFUNCTIONSUM^{2/3}(n, f, S_P^f)
    assume: f は乗法的関数である
    assume: S_P^f(v) = Σ_{i=1}^{π(v)} f(p_i)
    assume: 素数 p と正整数 e に対して f(p^e) が計算できる^a
    assume: 正整数 v ⩽ √n に対して S_P^f(v) が計算できる
    assume: 正整数 v ⩽ √n に対して S_P^f(⌊n/v⌋) が計算できる
    A = (n, ⌊n/2⌋, ..., ⌊n/⌊√n⌋⌋, ⌊n/⌊√n⌋⌋ − 1, ..., 2, 1)
    h ← (0)_{i=1}^{|A|}
    q = p_{π(n^{1/3})+1}
    foreach i ← (1, 2, ..., |A|) do
        m ← A_i
        h_i ←+ 1
        if m ⩽ n^{1/3} then continue
        h_i ←+ S_P^f(m) − S_P^f(q − 1)
        if m ⩽ n^{2/3} then continue
        foreach j ← (π(q), π(q) + 1, ...) do
            if p_j > ⌊m/p_j⌋ then break
            h_i ←+ f(p_j^2) + f(p_j) · (S_P^f(⌊m/p_j⌋) − S_P^f(p_j))
    b ← (0)_{i=⌊∛n⌋}^{|A|}
    π ← π(n^{1/3}) + 1
    while p_{π−1} > n^{1/6} do
        π ←− 1
        foreach i ← (1, ..., ⌊∛n⌋) do
            m ← A_i
            if m < p_π then break
            foreach e ← (1, 2, ...) do
                d = ⌊m/p_π^e⌋
                j = (if d ⩽ √n then |A| − d else ⌊n/d⌋)
                if h > ∛n then
                    h_i ←+ f(p_π^e) · (h_j + Σ_{k=j}^{|A|} b_k)
                else
                    h_i ←+ f(p_π^e) · h_j
                if m < p_π^{e+1} then break
            s ← ((p, π, 1, p^1))                ▷ s は stack で管理する。
            while (v, i, ϕ, p^e) ← s.POP() do
                j = (if v ⩽ √n then |A| − v else ⌊n/v⌋)
                if j > ∛n then b_j ←+ ϕ · f(p_i^e)
                if v · p_i < ⌊n/⌊∛n⌋⌋ then
                    s.PUSH((v · p_i, i, ϕ, p_i^{e+1}))
                for j ← (i + 1, ...) do
                    if v · p_j ⩾ ⌊n/⌊∛n⌋⌋ then break
                    s.PUSH((v · p_j, j, ϕ · f(p^e), p_j^1))
    foreach j ∈ (|A|, ..., ⌊∛n⌋) do
        h_j ←+ Σ_{k=j}^{|A|} b_k
    while p_{π−1} ⩾ 2 do
        π ←− 1
        foreach i ∈ (1, 2, ...) do
            m ← A_i
            if m < p then break
            foreach e ← (1, 2, ...) do
                d = ⌊m/p_π⌋
                j = (if d ⩽ √n then |A| − d else ⌊n/d⌋)
                h_i ←+ f(p^e) · h_j
                if m < p^{e+1} then break
    return h_1                                  ▷ Σ_{i=1}^{n} f(i)
```

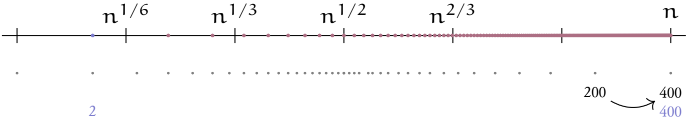
^a 与えられた p^e の値から p や e を高速に復元するのは難しいため、実際には (p, e, p^e) などの組の形で管理するのがよい場合もありそう。

高速化 Lucy DP の動き

Lucy DP を $\Theta(n^{2/3})$ 時間に高速化したアルゴリズムの動作。

$n = 400$ の場合を例として載せる。

$n^{1/6}$ 以下の素数で Lucy DP

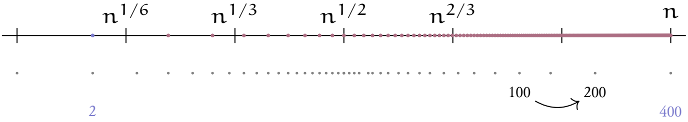


$$\underbrace{dp[1]}_{\substack{S_1(400)\\=399}} \overset{-}{\leftarrow} \underbrace{dp[2]}_{\substack{S_1(200)\\=199}} - \underbrace{\pi(2-1)}_{0}$$

$$dp[1] = S_2(400) = 200$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

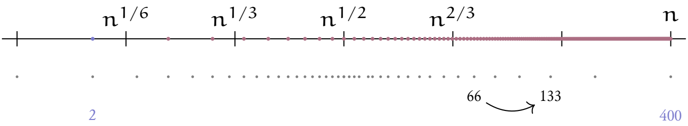


$$\underbrace{dp[2]}_{\substack{S_1(200)\\=199}} \overset{-}{\leftarrow} \underbrace{dp[4]}_{\substack{S_1(100)\\=99}} - \underbrace{\pi(2-1)}_{0}$$

$$dp[2] = S_2(200) = 100$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

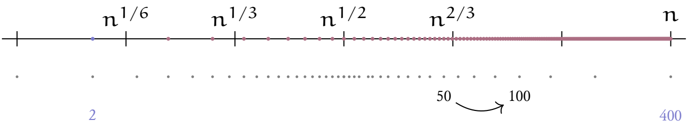


$$\underbrace{\mathrm{dp}[3]}_{\substack{S_1(133)\\=132}} \overset{-}{\longleftarrow} \underbrace{\mathrm{dp}[6]}_{\substack{S_1(66)\\=65}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[3] = S_2(133) = 67$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

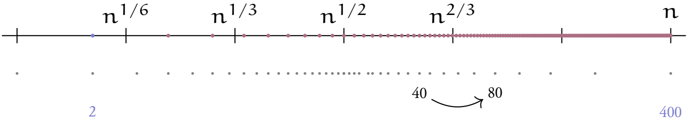


$$\underbrace{\mathrm{dp}[4]}_{\substack{S_1(100)\\=99}} \overset{-}{\longleftarrow} \underbrace{\mathrm{dp}[8]}_{\substack{S_1(50)\\=49}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[4] = S_2(100) = 50$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

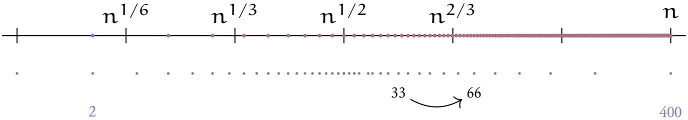


$$\underbrace{\mathrm{dp}[5]}_{\substack{S_1(80)\\=79}} \overset{-}{\leftarrow} \underbrace{\mathrm{dp}[10]}_{\substack{S_1(40)\\=39}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[5] = S_2(80) = 40$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

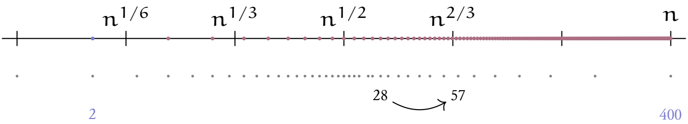


$$\underbrace{\mathrm{dp}[6]}_{\substack{S_1(66)\\=65}} \xleftarrow{-} \underbrace{\mathrm{dp}[12]}_{\substack{S_1(33)\\=32}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[6] = S_2(66) = 33$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

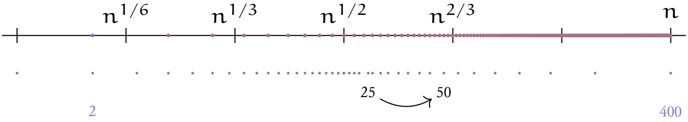


$$\underbrace{\mathrm{dp}[7]}_{\substack{S_1(57)\\=56}} \overset{-}{\leftarrow} \underbrace{\mathrm{dp}[14]}_{\substack{S_1(28)\\=27}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[7] = S_2(57) = 29$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

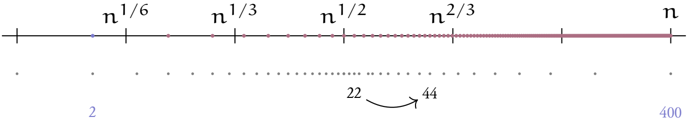


$$\underbrace{\mathrm{dp}[8]}_{\substack{S_1(50)\\=49}} \xleftarrow{-} \underbrace{\mathrm{dp}[16]}_{\substack{S_1(25)\\=24}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[8] = S_2(50) = 25$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

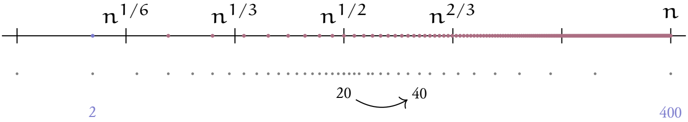


$$\underbrace{\mathrm{dp}[9]}_{\substack{S_1(44)\\=43}} \xleftarrow{-} \underbrace{\mathrm{dp}[18]}_{\substack{S_1(22)\\=21}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[9] = S_2(44) = 22$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

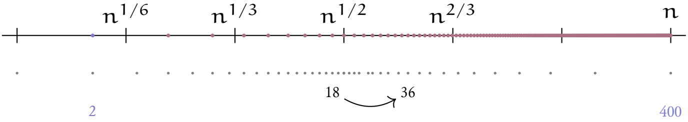


$$\underbrace{dp[10]}_{\substack{S_1(40)\\=39}} \overset{-}{\longleftarrow} \underbrace{dp[20]}_{\substack{S_1(20)\\=19}} - \underbrace{\pi(2-1)}_{0}$$

$$dp[10] = S_2(40) = 20$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

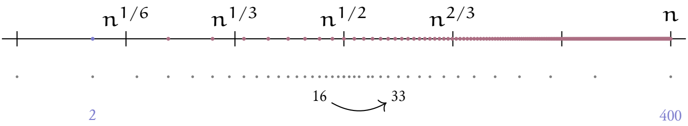


$$\underbrace{\mathrm{dp}[11]}_{\substack{S_1(36)\\=35}} \xleftarrow{-} \underbrace{\mathrm{dp}[22]}_{\substack{S_1(18)\\=17}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[11] = S_2(36) = 18$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

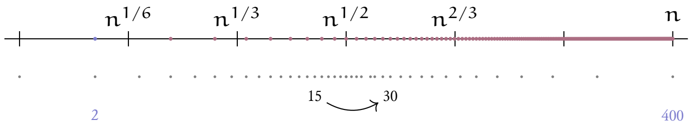


$$\underbrace{\mathrm{dp}[12]}_{\substack{S_1(33)\\=32}} \xleftarrow{-} \underbrace{\mathrm{dp}[24]}_{\substack{S_1(16)\\=15}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[12] = S_2(33) = 17$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

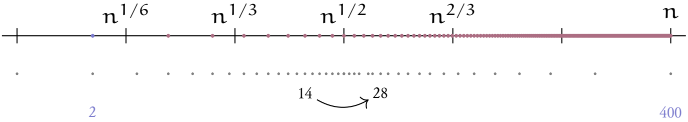


$$\underbrace{\mathrm{dp}[13]}_{\substack{S_1(30)\\=29}} \overset{-}{\leftarrow} \underbrace{\mathrm{dp}[25]}_{\substack{S_1(15)\\=14}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[13] = S_2(30) = 15$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

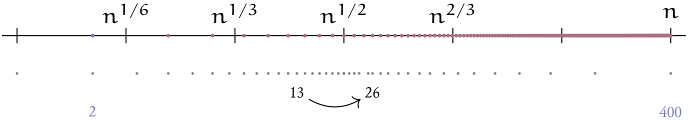


$$\underbrace{\mathrm{dp}[14]}_{\substack{S_1(28)\\=27}} \overset{-}{\longleftarrow} \underbrace{\mathrm{dp}[26]}_{\substack{S_1(14)\\=13}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[14] = S_2(28) = 14$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

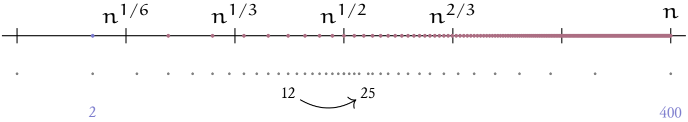


$$\underbrace{\mathrm{dp}[15]}_{\substack{S_1(26)\\=25}} \leftharpoondown \underbrace{\mathrm{dp}[27]}_{\substack{S_1(13)\\=12}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[15] = S_2(26) = 13$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

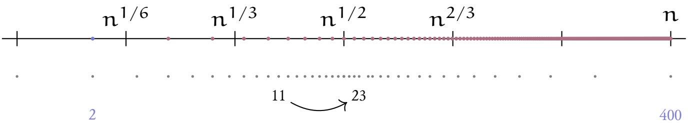


$$\underbrace{\mathrm{dp}[16]}_{\substack{S_1(25)\\=24}} \xleftarrow{-} \underbrace{\mathrm{dp}[28]}_{\substack{S_1(12)\\=11}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[16] = S_2(25) = 13$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

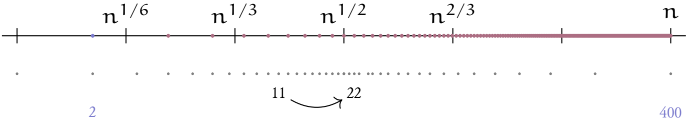


$$\underbrace{\mathrm{dp}[17]}_{\substack{S_1(23)\\=22}} \xleftarrow{-} \underbrace{\mathrm{dp}[29]}_{\substack{S_1(11)\\=10}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[17] = S_2(23) = 12$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

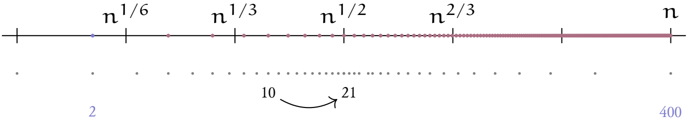


$$\underbrace{\mathrm{dp}[18]}_{\substack{S_1(22)\\=21}} \xleftarrow{-} \underbrace{\mathrm{dp}[29]}_{\substack{S_1(11)\\=10}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[18] = S_2(22) = 11$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

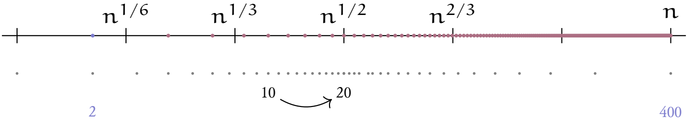


$$\underbrace{\mathrm{dp}[19]}_{\substack{S_1(21)\\=20}} \xleftarrow{-} \underbrace{\mathrm{dp}[30]}_{\substack{S_1(10)\\=9}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[19] = S_2(21) = 11$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

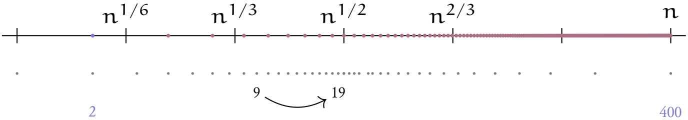


$$\underbrace{\mathrm{dp}[20]}_{\substack{S_1(20)\\=19}} \leftharpoondown \underbrace{\mathrm{dp}[30]}_{\substack{S_1(10)\\=9}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[20] = S_2(20) = 10$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

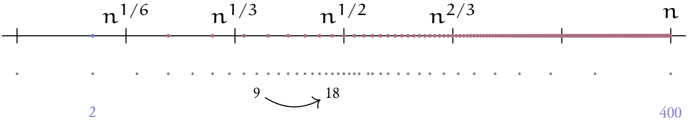


$$\underbrace{dp[21]}_{\substack{S_1(19)\\=18}} \overset{-}{\leftarrow} \underbrace{dp[31]}_{\substack{S_1(9)\\=8}} - \underbrace{\pi(2-1)}_{0}$$

$$dp[21] = S_2(19) = 10$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

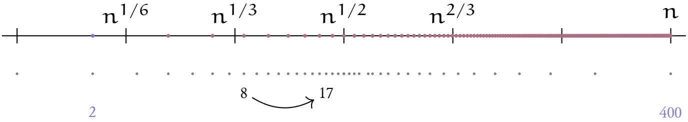


$$\underbrace{\mathrm{dp}[22]}_{\substack{S_1(18)\\=17}} \overset{-}{\leftarrow} \underbrace{\mathrm{dp}[31]}_{\substack{S_1(9)\\=8}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[22] = S_2(18) = 9$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

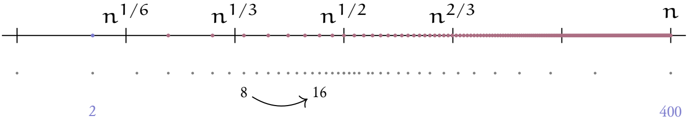


$$\underbrace{\mathrm{dp}[23]}_{\substack{S_1(17)\\=16}} \overset{-}{\leftarrow} \underbrace{\mathrm{dp}[32]}_{\substack{S_1(8)\\=7}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[23] = S_2(17) = 9$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

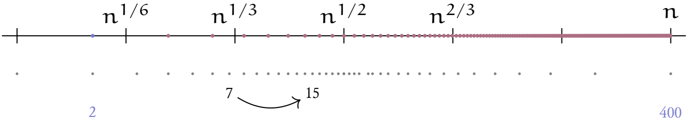


$$\underbrace{\mathrm{dp}[24]}_{\substack{S_1(16)\\=15}} \leftharpoonup \underbrace{\mathrm{dp}[32]}_{\substack{S_1(8)\\=7}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[24] = S_2(16) = 8$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

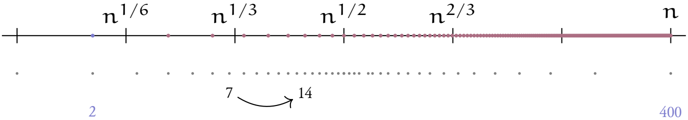


$$\underbrace{dp[25]}_{\substack{S_1(15)\\=14}} \overset{-}{\leftarrow} \underbrace{dp[33]}_{\substack{S_1(7)\\=6}} - \underbrace{\pi(2-1)}_{0}$$

$$dp[25] = S_2(15) = 8$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

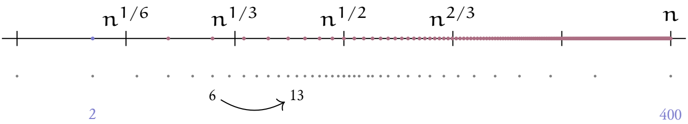


$$\underbrace{\mathrm{dp}[26]}_{\substack{S_1(14)\\=13}} \xleftarrow{-} \underbrace{\mathrm{dp}[33]}_{\substack{S_1(7)\\=6}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[26] = S_2(14) = 7$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

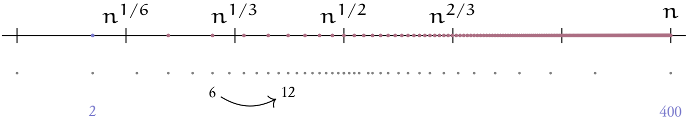


$$\underbrace{\mathrm{dp}[27]}_{\substack{S_1(13)\\=12}} \xleftarrow{-} \underbrace{\mathrm{dp}[34]}_{\substack{S_1(6)\\=5}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[27] = S_2(13) = 7$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

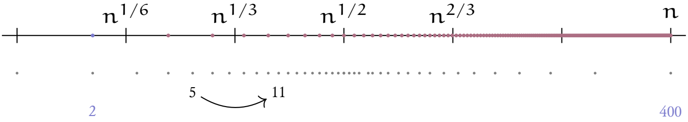


$$\underbrace{\mathrm{dp}[28]}_{\substack{S_1(12)\\=11}} \xleftarrow{-} \underbrace{\mathrm{dp}[34]}_{\substack{S_1(6)\\=5}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[28] = S_2(12) = 6$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

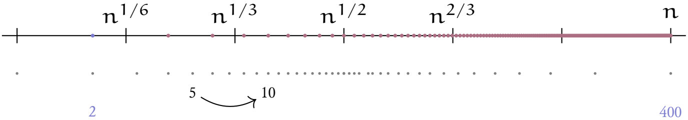


$$\underbrace{dp[29]}_{\substack{S_1(11)\\=10}} \overset{-}{\leftarrow} \underbrace{dp[35]}_{\substack{S_1(5)\\=4}} - \underbrace{\pi(2-1)}_{0}$$

$$dp[29] = S_2(11) = 6$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

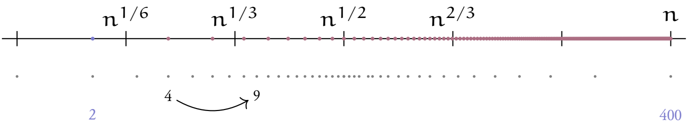


$$\underbrace{\mathrm{dp}[30]}_{\substack{S_1(10)\\=9}} \overset{-}{\leftarrow} \underbrace{\mathrm{dp}[35]}_{\substack{S_1(5)\\=4}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[30] = S_2(10) = 5$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

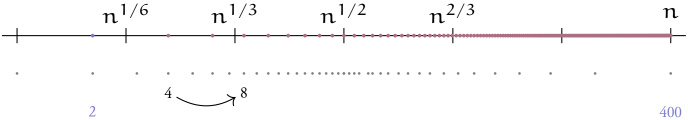


$$\underbrace{\mathrm{dp}[31]}_{\substack{S_1(9)\\=8}} \overset{-}{\leftarrow} \underbrace{\mathrm{dp}[36]}_{\substack{S_1(4)\\=3}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[31] = S_2(9) = 5$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

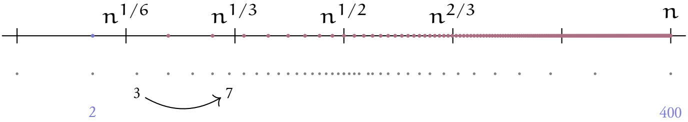


$$\underbrace{\mathrm{dp}[32]}_{\substack{S_1(8)\\=7}} \overset{-}{\leftarrow} \underbrace{\mathrm{dp}[36]}_{\substack{S_1(4)\\=3}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[32] = S_2(8) = 4$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

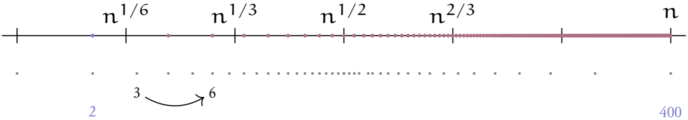


$$\underbrace{\mathrm{dp}[33]}_{\substack{S_1(7)\\=6}} \xleftarrow{-} \underbrace{\mathrm{dp}[37]}_{\substack{S_1(3)\\=2}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[33] = S_2(7) = 4$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

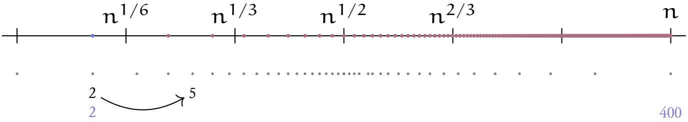


$$\underbrace{\mathrm{dp}[34]}_{\substack{S_1(6)\\=5}} \xleftarrow{-} \underbrace{\mathrm{dp}[37]}_{\substack{S_1(3)\\=2}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[34] = S_2(6) = 3$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

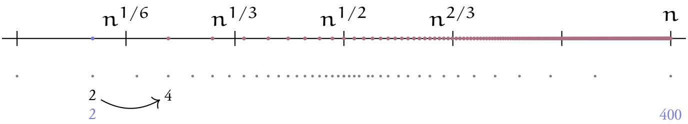


$$\underbrace{\mathrm{dp}[35]}_{\substack{S_1(5)\\=4}} \overset{-}{\leftarrow} \underbrace{\mathrm{dp}[38]}_{\substack{S_1(2)\\=1}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[35] = S_2(5) = 3$$

Figure: アルゴリズムの動き

$n^{1/6}$ 以下の素数で Lucy DP

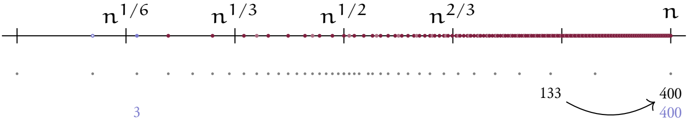


$$\underbrace{\mathrm{dp}[36]}_{\substack{S_1(4)\\=3}} \xleftarrow{-} \underbrace{\mathrm{dp}[38]}_{\substack{S_1(2)\\=1}} - \underbrace{\pi(2-1)}_{0}$$

$$\mathrm{dp}[36] = S_2(4) = 2$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

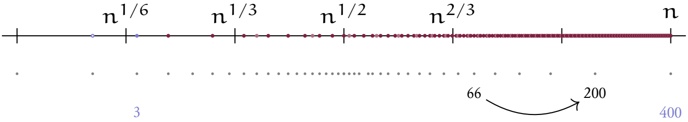


$$\underbrace{\mathrm{dp}[1]}_{\substack{S_2(400)\\=200}} \leftharpoonup \underbrace{\mathrm{dp}[3]}_{\substack{S_2(133)\\=67}} - \underbrace{\pi(3-1)}_{1}$$

$$\mathrm{dp}[1] = S_3(400) = 134$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

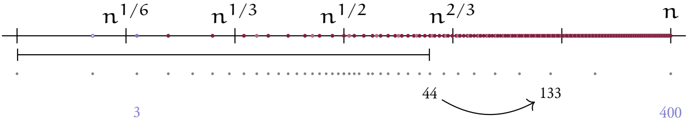


$$\underbrace{\mathrm{dp}[2]}_{\substack{S_2(200)\\=100}} \xleftarrow{-} \underbrace{\mathrm{dp}[6]}_{\substack{S_2(66)\\=33}} - \underbrace{\pi(3-1)}_{1}$$

$$\mathrm{dp}[2] = S_3(200) = 68$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

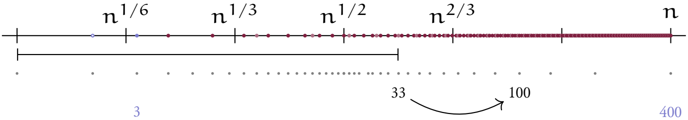


$$\underbrace{\mathrm{dp}[3]}_{\substack{S_2(133)\\=67}} \overset{-}{\longleftarrow} \underbrace{\mathrm{dp}[9]}_{\substack{S_2(44)\\=22}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 44,\ \sqrt[6]{n}<\mathrm{lpf}(v)<3\right\}\right|}_{\substack{|\{\}|\\=0}} - \underbrace{\pi(3-1)}_{1}$$

$$\mathrm{dp}[3] = S_3(133) = 46$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

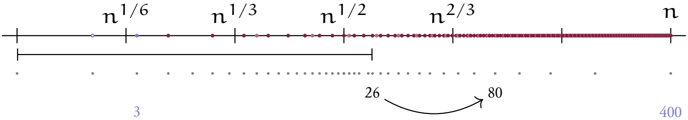


$$\underbrace{\mathrm{dp}[4]}_{\substack{S_2(100)\\=50}} \xleftarrow{-} \underbrace{\mathrm{dp}[12]}_{\substack{S_2(33)\\=17}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 33,\ \sqrt[6]{n}<\mathrm{lpf}(v)<3\right\}\right|}_{\substack{|\{\}|\\=0}} - \underbrace{\pi(3-1)}_{1}$$

$$\mathrm{dp}[4] = S_3(100) = 34$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

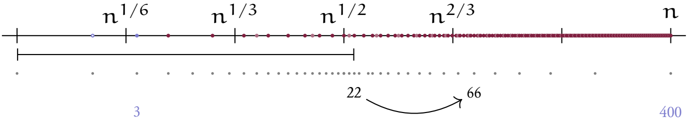


$$\underbrace{\mathrm{dp}[5]}_{\substack{S_2(80)\\=40}} \overset{-}{\longleftarrow} \underbrace{\mathrm{dp}[15]}_{\substack{S_2(26)\\=13}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 26,\ \sqrt[6]{n}<\mathrm{lpf}(v)<3\right\}\right|}_{\substack{|\{\}|\\=0}} - \underbrace{\pi(3-1)}_{1}$$

$$\mathrm{dp}[5] = S_3(80) = 28$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

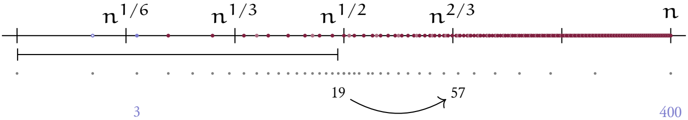


$$\underbrace{\mathrm{dp}[6]}_{\substack{S_2(66)\\=33}} \overset{-}{\longleftarrow} \underbrace{\mathrm{dp}[18]}_{\substack{S_2(22)\\=11}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 22,\ \sqrt[6]{n}<\mathrm{lpf}(v)<3\right\}\right|}_{\substack{|\{\}|\\=0}} - \underbrace{\pi(3-1)}_{1}$$

$$\mathrm{dp}[6] = S_3(66) = 23$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

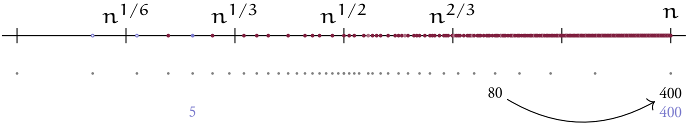


$$\underbrace{\mathrm{dp}[7]}_{\substack{S_2(57)\\=29}} \overset{-}{\longleftarrow} \underbrace{\mathrm{dp}[21]}_{\substack{S_2(19)\\=10}} - \underbrace{\left|\left\{v \in (\mathbb{N} \setminus \mathbb{P}) \mid v \leqslant 19,\ \sqrt[6]{n} < \mathrm{lpf}(v) < 3\right\}\right|}_{\substack{|\{\}|\\=0}} - \underbrace{\pi(3-1)}_{1}$$

$$\mathrm{dp}[7] = S_3(57) = 20$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

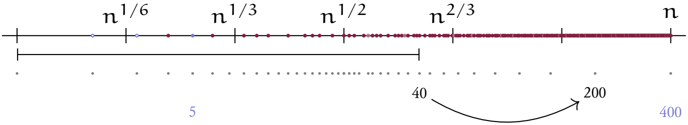


$$\underbrace{\mathrm{dp}[1]}_{\substack{S_3(400)\\=134}} \leftharpoondown \underbrace{\mathrm{dp}[5]}_{\substack{S_3(80)\\=28}} - \underbrace{\pi(5-1)}_{2}$$

$$\mathrm{dp}[1] = S_5(400) = 108$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

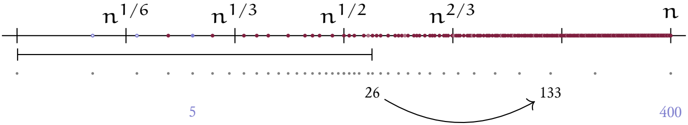


$$\underbrace{\mathrm{dp}[2]}_{\substack{S_3(200)\\=68}} \xleftarrow{-} \underbrace{\mathrm{dp}[10]}_{\substack{S_2(40)\\=20}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 40,\ \sqrt[6]{n}<\mathrm{lpf}(v)<5\right\}\right|}_{\substack{|\{9,15,21,27,33,39\}|\\=6}} - \underbrace{\pi(5-1)}_{2}$$

$$\mathrm{dp}[2] = S_5(200) = 56$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

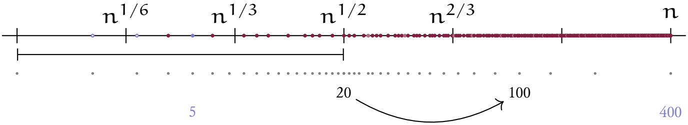


$$\underbrace{\mathrm{dp}[3]}_{\substack{S_3(133)\\=46}} \leftarrow \underbrace{\mathrm{dp}[15]}_{\substack{S_2(26)\\=13}} - \underbrace{\left|\left\{\nu\in(\mathbb{N}\setminus\mathbb{P}) \mid \nu\leqslant 26,\ \sqrt[6]{n}<\mathrm{lpf}(\nu)<5\right\}\right|}_{\substack{|\{9,15,21\}|\\=3}} - \underbrace{\pi(5-1)}_{2}$$

$$\mathrm{dp}[3] = S_5(133) = 38$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

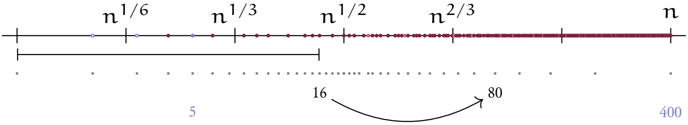


$$\underbrace{\mathrm{dp}[4]}_{\substack{S_3(100)\\=34}} \leftarrow \underbrace{\mathrm{dp}[20]}_{\substack{S_2(20)\\=10}} - \underbrace{\left|\left\{v \in (\mathbb{N}\setminus\mathbb{P}) \mid v \leqslant 20,\ \sqrt[6]{n} < \mathrm{lpf}(v) < 5\right\}\right|}_{\substack{|\{9,15\}|\\=2}} - \underbrace{\pi(5-1)}_{2}$$

$$\mathrm{dp}[4] = S_5(100) = 28$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

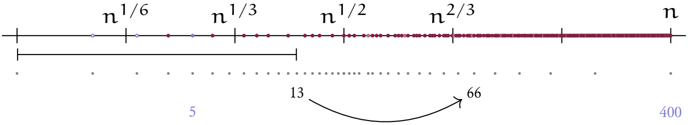


$$\underbrace{\mathrm{dp}[5]}_{\substack{S_3(80)\\=28}} \overset{-}{\longleftarrow} \underbrace{\mathrm{dp}[24]}_{\substack{S_2(16)\\=8}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 16,\ \sqrt[6]{n}<\mathrm{lpf}(v)<5\right\}\right|}_{\substack{|\{9,15\}|\\=2}} - \underbrace{\pi(5-1)}_{2}$$

$$\mathrm{dp}[5] = S_5(80) = 24$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

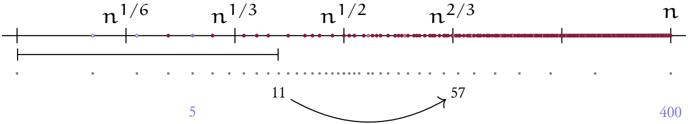


$$\underbrace{\mathrm{dp}[6]}_{\substack{S_3(66)\\=23}} \overset{-}{\longleftarrow} \underbrace{\mathrm{dp}[27]}_{\substack{S_2(13)\\=7}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 13,\ \sqrt[6]{n}<\mathrm{lpf}(v)<5\right\}\right|}_{\substack{|\{9\}|\\=1}} - \underbrace{\pi(5-1)}_{2}$$

$$\mathrm{dp}[6] = S_5(66) = 19$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

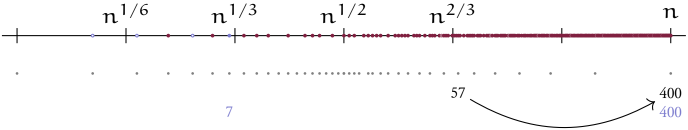


$$\underbrace{\mathrm{dp}[7]}_{\substack{S_3(57)\\=20}} \xleftarrow{-} \underbrace{\mathrm{dp}[29]}_{\substack{S_2(11)\\=6}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 11,\ \sqrt[6]{n}<\mathrm{lpf}(v)<5\right\}\right|}_{\substack{|\{9\}|\\=1}} - \underbrace{\pi(5-1)}_{2}$$

$$\mathrm{dp}[7] = S_5(57) = 17$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

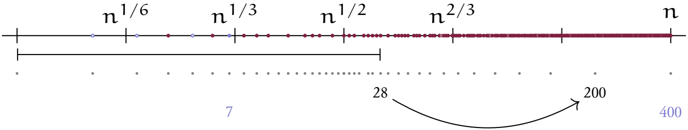


$$\underbrace{\mathrm{dp}[1]}_{\substack{S_5(400)\\=108}} \leftharpoonup \underbrace{\mathrm{dp}[7]}_{\substack{S_5(57)\\=17}} - \underbrace{\pi(7-1)}_{3}$$

$$\mathrm{dp}[1] = S_7(400) = 94$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

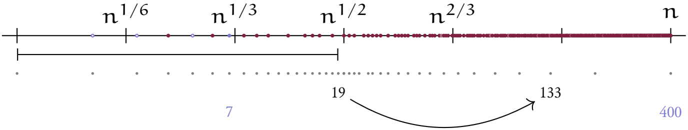


$$\underbrace{\mathrm{dp}[2]}_{\substack{S_5(200)\\=56}} \longleftarrow \underbrace{\mathrm{dp}[14]}_{\substack{S_2(28)\\=14}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 28,\ \sqrt[6]{n}<\mathrm{lpf}(v)<7\right\}\right|}_{\substack{|\{9,15,21,25,27\}|\\=5}} - \underbrace{\pi(7-1)}_{3}$$

$$\mathrm{dp}[2] = S_7(200) = 50$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

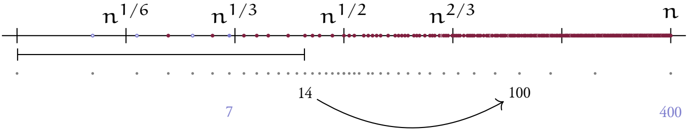


$$\underbrace{\mathrm{dp}[3]}_{\substack{S_5(133)\\=38}} \leftarrow \underbrace{\mathrm{dp}[21]}_{\substack{S_2(19)\\=10}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 19,\ \sqrt[6]{n}<\mathrm{lpf}(v)<7\right\}\right|}_{\substack{|\{9,15\}|\\=2}} - \underbrace{\pi(7-1)}_{3}$$

$$\mathrm{dp}[3] = S_7(133) = 33$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

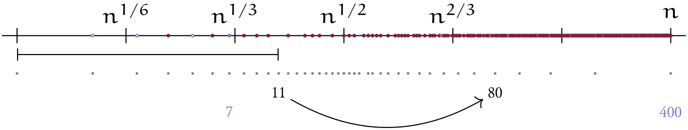


$$\underbrace{\mathrm{dp}[4]}_{\substack{S_5(100)\\=28}} \xleftarrow{-} \underbrace{\mathrm{dp}[26]}_{\substack{S_2(14)\\=7}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 14,\ \sqrt[6]{n}<\mathrm{lpf}(v)<7\right\}\right|}_{\substack{|\{9\}|\\=1}} - \underbrace{\pi(7-1)}_{3}$$

$$\mathrm{dp}[4] = S_7(100) = 25$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

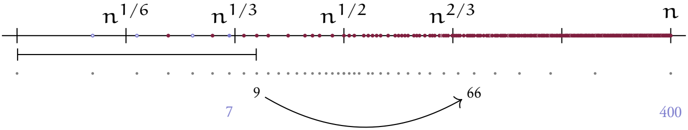


$$\underbrace{\mathrm{dp}[5]}_{\substack{S_5(80)\\=24}} \overset{-}{\longleftarrow} \underbrace{\mathrm{dp}[29]}_{\substack{S_2(11)\\=6}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 11,\ \sqrt[6]{n}<\mathrm{lpf}(v)<7\right\}\right|}_{\substack{|\{9\}|\\=1}} - \underbrace{\pi(7-1)}_{3}$$

$$\mathrm{dp}[5] = S_7(80) = 22$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

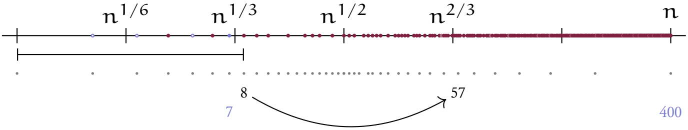


$$\underbrace{\mathrm{dp}[6]}_{\substack{S_5(66)\\=19}} \overset{-}{\longleftarrow} \underbrace{\mathrm{dp}[31]}_{\substack{S_2(9)\\=5}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 9,\ \sqrt[6]{n}<\mathrm{lpf}(v)<7\right\}\right|}_{\substack{|\{9\}|\\=1}} - \underbrace{\pi(7-1)}_{3}$$

$$\mathrm{dp}[6] = S_7(66) = 18$$

Figure: アルゴリズムの動き

$n^{1/3}$ 未満の素数で Lucy DP + 合成数列挙

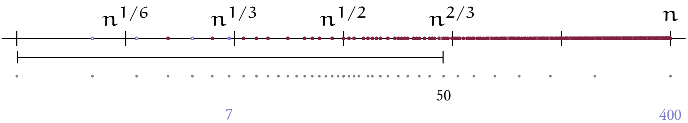


$$\underbrace{\mathrm{dp}[7]}_{\substack{S_5(57)\\=17}} \leftarrow \underbrace{\mathrm{dp}[32]}_{\substack{S_2(8)\\=4}} - \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 8,\ \sqrt[6]{n}<\mathrm{lpf}(v)<7\right\}\right|}_{\substack{|\{\}|\\=0}} - \underbrace{\pi(7-1)}_{3}$$

$$\mathrm{dp}[7] = S_7(57) = 16$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

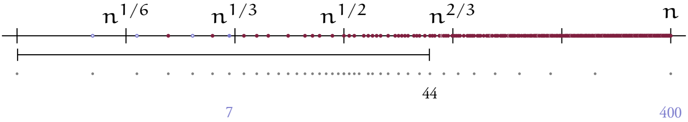


$$\underbrace{\mathrm{dp}[8]}_{\substack{S_2(50)\\=25}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 50,\ \sqrt[6]{n}<\operatorname{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15,21,25,27,33,35,39,45,49\}|\\=10}}$$

$$\mathrm{dp}[8] = S_7(50) = 15$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

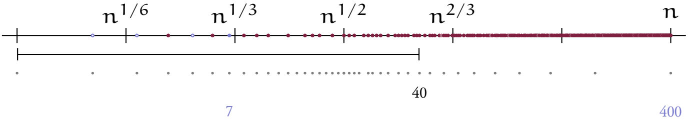


$$\underbrace{\mathrm{dp}[9]}_{\substack{S_2(44)\\=22}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 44,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15,21,25,27,33,35,39\}|\\=8}}$$

$$\mathrm{dp}[9] = S_7(44) = 14$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

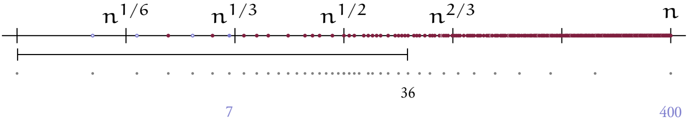


$$\underbrace{\mathrm{dp}[10]}_{\substack{S_2(40)\\=20}} \overset{-}{\longleftarrow} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 40,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15,21,25,27,33,35,39\}|\\=8}}$$

$$\mathrm{dp}[10] = S_7(40) = 12$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

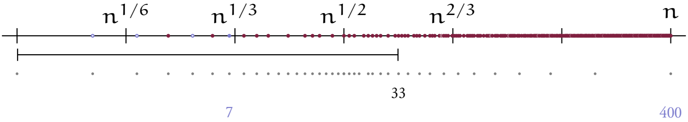


$$\underbrace{\mathrm{dp}[11]}_{\substack{S_2(36)\\=18}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 36,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15,21,25,27,33,35\}|\\=7}}$$

$$\mathrm{dp}[11] = S_7(36) = 11$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

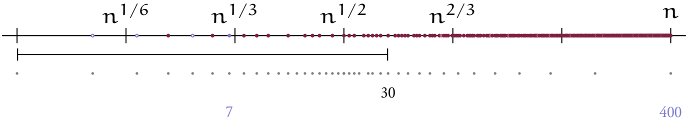


$$\underbrace{\mathrm{dp}[12]}_{\substack{S_2(33)\\=17}} \longleftarrow \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 33,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15,21,25,27,33\}|\\=6}}$$

$$\mathrm{dp}[12] = S_7(33) = 11$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

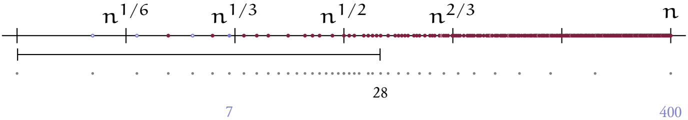


$$\underbrace{\mathrm{dp}[13]}_{\substack{S_2(30)\\=15}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 30,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15,21,25,27\}|\\=5}}$$

$$\mathrm{dp}[13] = S_7(30) = 10$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

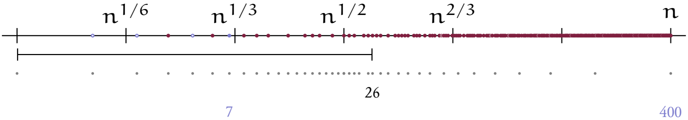


$$\underbrace{\mathrm{dp}[14]}_{\substack{S_2(28)\\=14}} \overset{-}{\longleftarrow} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 28,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15,21,25,27\}|\\=5}}$$

$$\mathrm{dp}[14] = S_7(28) = 9$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

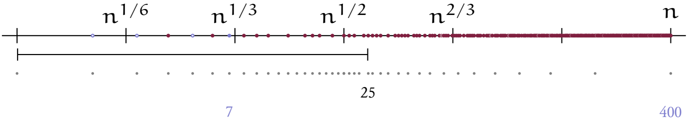


$$\underbrace{\mathrm{dp}[15]}_{\substack{S_2(26)\\=13}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 26,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15,21,25\}|\\=4}}$$

$$\mathrm{dp}[15] = S_7(26) = 9$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

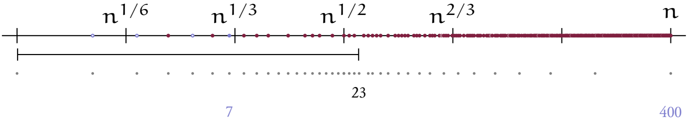


$$\underbrace{\mathrm{dp}[16]}_{\substack{S_2(25)\\=13}} \overset{-}{\longleftarrow} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 25,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15,21,25\}|\\=4}}$$

$$\mathrm{dp}[16] = S_7(25) = 9$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

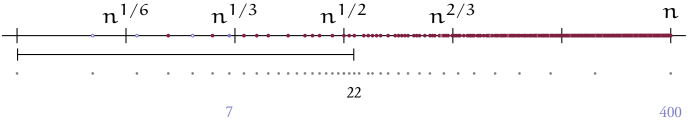


$$\underbrace{\mathrm{dp}[17]}_{\substack{S_2(23)\\=12}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 23,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15,21\}|\\=3}}$$

$$\mathrm{dp}[17] = S_7(23) = 9$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

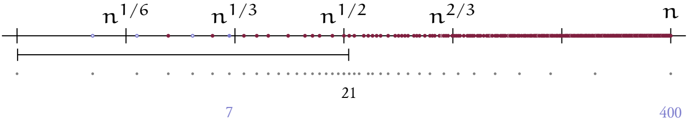


$$\underbrace{\mathrm{dp}[18]}_{\substack{S_2(22)\\=11}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 22,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15,21\}|\\=3}}$$

$$\mathrm{dp}[18] = S_7(22) = 8$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

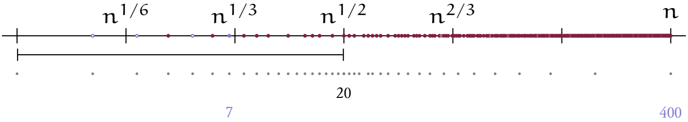


$$\underbrace{\mathrm{dp}[19]}_{\substack{S_2(21)\\=11}} \overset{-}{\longleftarrow} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 21,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15,21\}|\\=3}}$$

$$\mathrm{dp}[19] = S_7(21) = 8$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

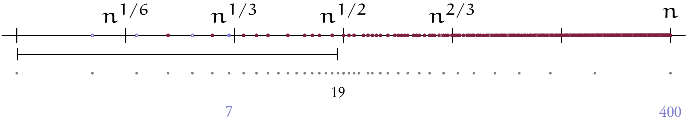


$$\underbrace{\mathrm{dp}[20]}_{\substack{S_2(20)\\=10}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 20,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15\}|\\=2}}$$

$$\mathrm{dp}[20] = S_7(20) = 8$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

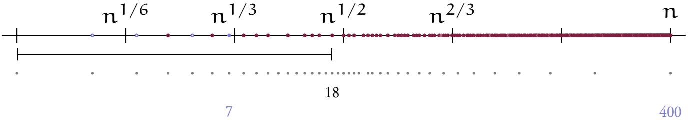


$$\underbrace{\mathrm{dp}[21]}_{\substack{S_2(19)\\=10}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 19,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15\}|\\=2}}$$

$$\mathrm{dp}[21] = S_7(19) = 8$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

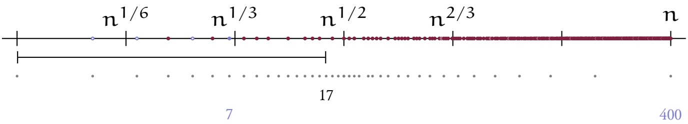


$$\underbrace{\mathrm{dp}[22]}_{\substack{S_2(18)\\=9}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 18,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15\}|\\=2}}$$

$$\mathrm{dp}[22] = S_7(18) = 7$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

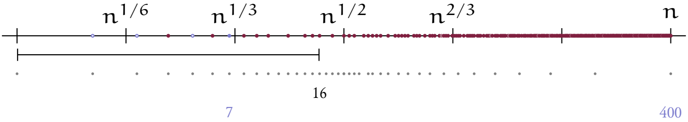


$$\underbrace{\mathrm{dp}[23]}_{\substack{S_2(17)\\=9}} \longleftarrow \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 17,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15\}|\\=2}}$$

$$\mathrm{dp}[23] = S_7(17) = 7$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

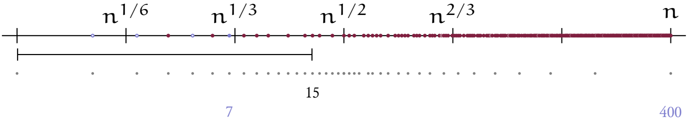


$$\underbrace{\mathrm{dp}[24]}_{\substack{S_2(16)\\=8}} \overset{-}{\longleftarrow} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 16,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15\}|\\=2}}$$

$$\mathrm{dp}[24] = S_7(16) = 6$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

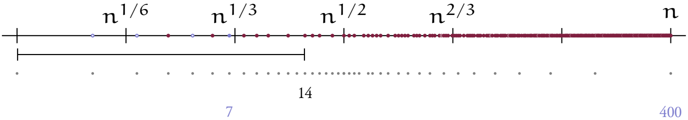


$$\underbrace{\mathrm{dp}[25]}_{\substack{S_2(15)\\=8}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 15,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9,15\}|\\=2}}$$

$$\mathrm{dp}[25] = S_7(15) = 6$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

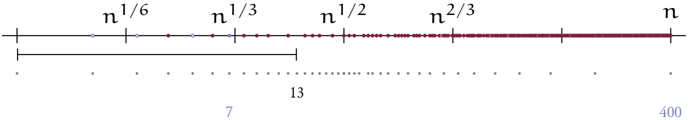


$$\underbrace{\mathrm{dp}[26]}_{S_2(14)=7} \longleftarrow \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 14,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{|\{9\}|=1}$$

$$\mathrm{dp}[26] = S_7(14) = 6$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

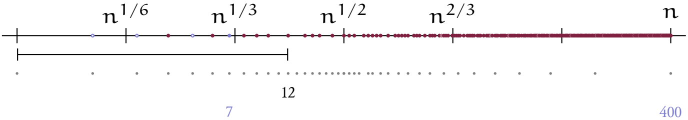


$$\underbrace{\mathrm{dp}[27]}_{\substack{S_2(13)\\=7}} \overset{-}{\longleftarrow} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 13,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9\}|\\=1}}$$

$$\mathrm{dp}[27] = S_7(13) = 6$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

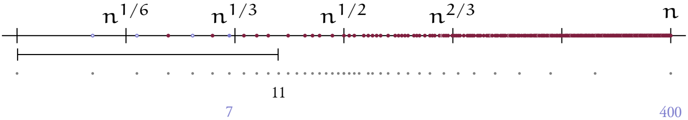


$$\underbrace{\mathrm{dp}[28]}_{\substack{S_2(12)\\=6}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 12,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9\}|\\=1}}$$

$$\mathrm{dp}[28] = S_7(12) = 5$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

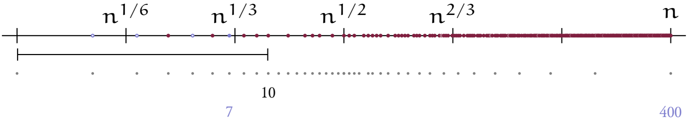


$$\underbrace{\mathrm{dp}[29]}_{\substack{S_2(11)\\=6}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 11,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9\}|\\=1}}$$

$$\mathrm{dp}[29] = S_7(11) = 5$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

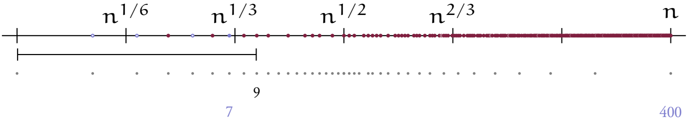


$$\underbrace{\mathrm{dp}[30]}_{\substack{S_2(10)\\=5}} \longleftarrow \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 10,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9\}|\\=1}}$$

$$\mathrm{dp}[30] = S_7(10) = 4$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

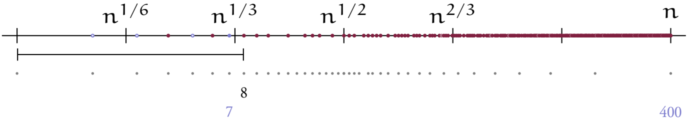


$$\underbrace{\mathrm{dp}[31]}_{\substack{S_2(9)\\=5}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 9,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{9\}|\\=1}}$$

$$\mathrm{dp}[31] = S_7(9) = 4$$

Figure: アルゴリズムの動き

列挙した合成数を DP 配列に反映

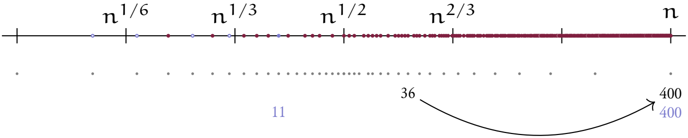


$$\underbrace{\mathrm{dp}[32]}_{\substack{S_2(8)\\=4}} \xleftarrow{-} \underbrace{\left|\left\{v\in(\mathbb{N}\setminus\mathbb{P}) \mid v\leqslant 8,\ \sqrt[6]{n}<\mathrm{lpf}(v)<n^{2/3}\right\}\right|}_{\substack{|\{\}|\\=0}}$$

$$\mathrm{dp}[32] = S_7(8) = 4$$

Figure: アルゴリズムの動き

$n^{1/2}$ 以下の素数で Lucy DP

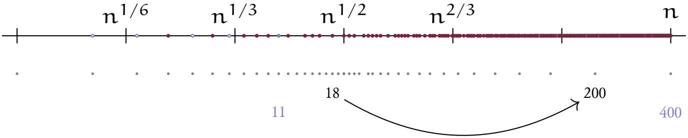


$$\underbrace{\mathrm{dp}[1]}_{\substack{S_7(400)\\=94}} \overset{-}{\longleftarrow} \underbrace{\mathrm{dp}[11]}_{\substack{S_7(36)\\=11}} - \underbrace{\pi(11-1)}_{4}$$

$$\mathrm{dp}[1] = S_{11}(400) = 87$$

Figure: アルゴリズムの動き

$n^{1/2}$ 以下の素数で Lucy DP

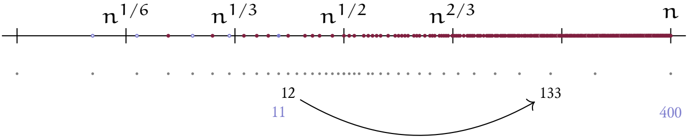


$$\underbrace{\mathrm{dp}[2]}_{\substack{S_7(200)\\=50}} \overset{-}{\leftarrow} \underbrace{\mathrm{dp}[22]}_{\substack{S_7(18)\\=7}} - \underbrace{\pi(11-1)}_{4}$$

$$\mathrm{dp}[2] = S_{11}(200) = 47$$

Figure: アルゴリズムの動き

$n^{1/2}$ 以下の素数で Lucy DP

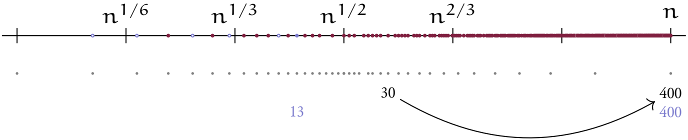


$$\underbrace{\mathrm{dp}[3]}_{\substack{S_7(133)\\=33}} \overset{-}{\leftarrow} \underbrace{\mathrm{dp}[28]}_{\substack{S_7(12)\\=5}} - \underbrace{\pi(11-1)}_{4}$$

$$\mathrm{dp}[3] = S_{11}(133) = 32$$

Figure: アルゴリズムの動き

$n^{1/2}$ 以下の素数で Lucy DP

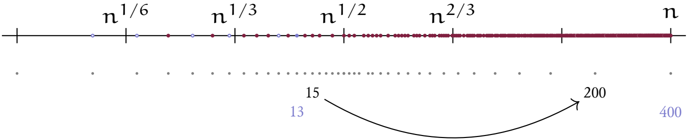


$$\underbrace{dp[1]}_{\substack{S_{11}(400)\\=87}} \overset{-}{\leftarrow} \underbrace{dp[13]}_{\substack{S_{11}(30)\\=10}} - \underbrace{\pi(13-1)}_{5}$$

$$dp[1] = S_{13}(400) = 82$$

Figure: アルゴリズムの動き

$n^{1/2}$ 以下の素数で Lucy DP

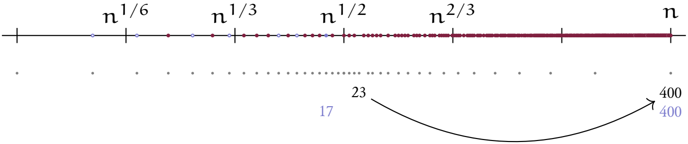


$$\underbrace{\mathrm{dp}[2]}_{\substack{S_{11}(200)\\=47}} \overset{-}{\leftarrow} \underbrace{\mathrm{dp}[25]}_{\substack{S_{11}(15)\\=6}} - \underbrace{\pi(13-1)}_{5}$$

$$\mathrm{dp}[2] = S_{13}(200) = 46$$

Figure: アルゴリズムの動き

$n^{1/2}$ 以下の素数で Lucy DP

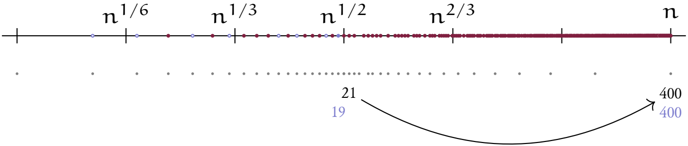


$$\underbrace{\mathrm{dp}[1]}_{\substack{S_{13}(400)\\=82}} \xleftarrow{-} \underbrace{\mathrm{dp}[17]}_{\substack{S_{13}(23)\\=9}} - \underbrace{\pi(17-1)}_{6}$$

$$\mathrm{dp}[1] = S_{17}(400) = 79$$

Figure: アルゴリズムの動き

$n^{1/2}$ 以下の素数で Lucy DP



$$\underbrace{dp[1]}_{\substack{S_{17}(400)\\=79}} \overset{-}{\leftarrow} \underbrace{dp[19]}_{\substack{S_{17}(21)\\=8}} - \underbrace{\pi(19-1)}_{7}$$

$$dp[1] = S_{19}(400) = 78$$

Figure: アルゴリズムの動き

おわり

$p_{\pi(\sqrt{400})} = 19$ であり、$S_{19}(400) = \pi(400) = 78$ が求められた。

おまけ

約数の総和を求める関数 σ_1 について、

$$\sigma_1\left(\prod_{p\colon \text{prime}} p^{e_p}\right) = \prod_{p\colon \text{prime}} \frac{p^{e_p+1}-1}{p-1}$$

が成り立つ。特に、$\sigma_1(p) = p + 1$ である。よって、今回の手法で各整数の "約数の和" の総和は $\Theta(n^{2/3})$ 時間で得られる[30]。

[30] ハンマーを持つとすべてが釘に見えるが、これは $O(\sqrt{n})$ 時間で解ける。
各約数 j の寄与を考える。n 以下の正整数のうち、j の倍数は $\lfloor n/j \rfloor$ 個あり、答えへの寄与は $\lfloor n/j \rfloor \cdot j$ となる。そのため、$\sum_{j=1}^{n} \lfloor n/j \rfloor \cdot j$ を求めればよいが、これは harmonic floor sum (fraction harmonic sum) と同じ方針で解ける。