Summing $\mu(n)$: a faster elementary algorithm

Lola Thompson

(joint with Harald Andrés Helfgott)

ANTS - XV

Universiteit Utrecht

The Mertens Function

Def The function

$$M(N) = \sum_{n \leq N} \mu(n)$$

is called the **Mertens Function**.

Ex $M(6) = \mu(1) + \mu(2) + \mu(3) + \mu(4) + \mu(5) + \mu(6)$

$= 1 + (-1) + (-1) + 0 + (-1) + 1$

$= \boxed{-1}$

Why Compute $M(N)$?

* Testing conjectures

  E.g., Until when is $|M(N)| \leq \sqrt{N}$ true?

  ↳ Known to be false for extremely large N (Odlyzko-te Riele)

* For N large, asymptotic expressions for $M(N)$ are good. For bounded N, they are not (Computations are preferable).

Naive Approach:

Compute $\mu(n)\ \forall n \leq N$ and sum.

Time: at least

$\Theta(N \cdot (\text{avg. time it takes to compute } \mu(n)))$

To save time, we can use the Sieve of Eratosthenes:

Time: $\Theta(N \log \log N)$

Space: $\Theta(N)$

To save space, we can use a segmented sieve:

70	71	72	73	74	75	76	77	78	79

Look at segments of length $\sqrt{N}$ and check for divisibility by primes up to $\sqrt{N}$.

Space: $\Theta(\sqrt{N})$

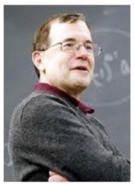
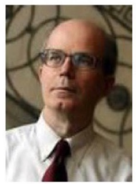
One more way to save space...

<u>Theorem</u> (Helfgott, 2020)

One can construct all primes $p \leq N$ in time $\Theta(N \log N)$ and space $\Theta(N^{1/3}(\log N)^{2/3})$.

Less Naive Methods

① Combinatorial Methods

First Steps: Meissel (1870s), Lehmer (1959)

Improved by Lagarias - Miller - Odlyzko (1985)

& Deléglise - Rivat (1996)

Main idea:

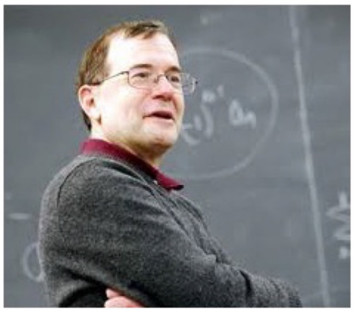
Use number theoretical identities to break

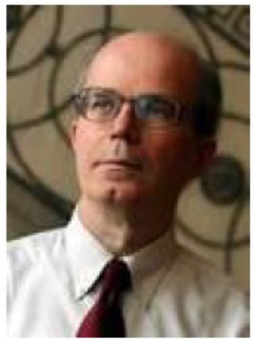
$$M(N) = \sum_{n \le N} \mu(n)$$

into shorter sums. Compute the short sums once & use them many times.

Time: about $O(N^{2/3})$

② Analytic Methods





Lagarias-Odlyzko (1987)

Main idea:

Can write $M(N)$ as sums over the zeros of the Riemann Zeta function. There are ∞ly many zeros, but one can truncate & round. If error $< \frac{1}{2}$, result is exact.

Time: $O(N^{1/2+\varepsilon})$ ← for $\pi(x)$ in theory

(nontrivial to implement (Platt, 2012) & slower than combinatorial methods in practice)

Our Work

Our Goal:

Formulate a combinatorial algorithm that

* improves on the previous time bound of $O(N^{2/3})$
* uses as little space as possible
* is practical to implement on a computer.

Theorem (Helfgott, T., 2021)

One can compute $M(N)$ in time $O_\varepsilon\left(N^{3/5}(\log N)^{3/5+\varepsilon}\right)$ and space $O\left(N^{3/10}(\log N)^{13/10}\right)$

$O(N^{3/5}\log N)$ using Helfgott's Sieve

$O\left(N^{1/5}(\log N)^{5/3}\right)$ using Helfgott's Sieve

Combinatorial algorithms: a general approach

Start w/ an identity:

$$M(N) = 2M(\sqrt{N}) - \sum_{n \le N} \sum_{\substack{m_1 m_2 | N \\ m_1, m_2 \le \sqrt{N}}} \mu(m_1)\mu(m_2)$$

Heath-Brown
$k=2$ case

Swapping the order of summation:

$$M(N) = \underbrace{2M(\sqrt{N})}_{\text{Compute naively in time } O(\sqrt{N})} - \boxed{\sum_{m_1, m_2 \le \sqrt{N}} \mu(m_1)\mu(m_2) \left\lfloor \frac{N}{m_1 m_2} \right\rfloor}$$

Choose a parameter $v \le \sqrt{N}$ and split into cases:

① $m_1, m_2 \le v$

② m_1 or $m_2 > v$

To obtain time $O(N^{2/3})$: ← Deleglise-Rivat, Lagarias-Miller-Odlyzko, etc.

Take $v = N^{1/3}$.

- Case ① ($m_1, m_2 \le v$) is the easy case – use a segmented sieve.
- Case ② (m_1 or $m_2 > v$) takes more work.

What we do instead:

take $\boxed{\nu \approx N^{2/5}}$

Larger $\nu \Rightarrow$ Case 1 is the hard case now.

This will be the focus of the rest of my talk

Case 2 is easier.

How we handle Case ①

Want to compute:

$$\sum \mu(m_1)\mu(m_2)\left\lfloor \frac{N}{m_1 m_2} \right\rfloor$$

m n from now on

Case ① $\rightarrow$ $m_1, m_2 \le v$

Spit $[1,v] \times [1,v]$ into nbhds

$U = I_x \times I_y$ around points (m_0, n_0)

$[m_0-a, m_0+a)$ $[n_0-b, n_0+b)$

Applying a local linear approximation:

$$\frac{N}{mn} = \frac{N}{m_0 n_0} + c_x(m-m_0) + c_y(n-n_0) + ET_{\text{Quad}},$$

small provided that U is small

where $c_x = \frac{-N}{m_0^2 n_0}$,

$$c_y = \frac{-N}{m_0 n_0^2}$$

If there were no floor functions...

and no ETQuad

$$\sum_{(m,n)\in I_x \times I_y} \mu(m)\mu(n)\frac{N}{mn}$$

$$= \sum_{(m,n)\in I_x \times I_y} \mu(m)\mu(n)\left(\frac{N}{m_0 n_0} + c_x(m-m_0) + c_y(n-n_0)\right)$$

$$\circledast = \sum_{m\in I_x} \mu(m)\left(\frac{N}{m_0 n_0} + c_x(m-m_0)\right)\cdot \sum_{n\in I_y}\mu(n)$$

$$+\left(\sum_{n\in I_y} \mu(n)\, c_y(n-n_0)\right)\cdot \sum_{m\in I_x}\mu(m)$$

* Use a segmented sieve to compute $\mu(m)$ for $m\in I_x$ and $\mu(n)$ for $n\in I_y$

* Computing the sum $\circledast$ takes time

$\Theta(\max(a,b))$

and negligible space.

How to handle $\lfloor\ \rfloor$

Notice that computing

Variables are separated

$$S_0 := \sum_{(m,n)\in I_x \times I_y} \mu(m)\mu(n)\left(\left\lfloor \frac{N}{m_0 n_0} + c_x(m-m_0)\right\rfloor + \left\lfloor c_y(n-n_0)\right\rfloor\right)$$

is the same as above.

what we have from our Linear Approx:

$$S_1 := \sum_{(m,n)\in I_x \times I_y} \mu(m)\mu(n)\left(\left\lfloor \frac{N}{m_0 n_0} + c_x(m-m_0) + c_y(n-n_0)\right\rfloor\right)$$

Can't be separated

what we actually want:

$$S_2 := \sum_{(m,n)\in I_x \times I_y} \mu(m)\mu(n)\left\lfloor \frac{N}{mn}\right\rfloor$$

Notice that

$$\lfloor A+B \rfloor - (\lfloor A \rfloor + \lfloor B \rfloor) = \begin{cases} 0 \text{ if } \{A\}+\{B\}<1 \\ 1 \text{ otw} \end{cases}$$

So the difference between a term in S_1 & a term in S_0 is either 0 or 1.

(Same for the terms in S_2 vs S_1)

<u>Idea</u>: Let

$$L_0(m,n) = \left\lfloor \frac{N}{m_0 n_0} + c_x(m-m_0) \right\rfloor + \lfloor c_y(n-n_0) \rfloor$$

$$L_1(m,n) = \left\lfloor \frac{N}{m_0 n_0} + c_x(m-m_0) + c_y(n-n_0) \right\rfloor$$

$$L_2(m,n) = \left\lfloor \frac{N}{mn} \right\rfloor$$

* We show that $L_2 - L_1$ and $L_1 - L_0$ can be computed quickly

We approximate c_y by a rational #

On each nbhd $I_x \times I_y$

$$\frac{a_0}{q}, \quad q \le Q = 2b$$

Such that $\delta := c_y - \frac{a_0}{q}$ satisfies

$$|\delta| \le \frac{1}{qQ}.$$

Thus,

$$\left| c_y(n-n_0) - \frac{a_0(n-n_0)}{q} \right| \le \frac{1}{2q}$$

* We can find such an $\frac{a_0}{q}$ in time $\Theta(\log b)$ using continued fractions

* Now, our task is to show that $L_2(m,n) = L_1(m,n)$ except in at most 2 "bad" congruence classes

(Same for $L_1(m,n)$ vs $L_0(m,n)$)

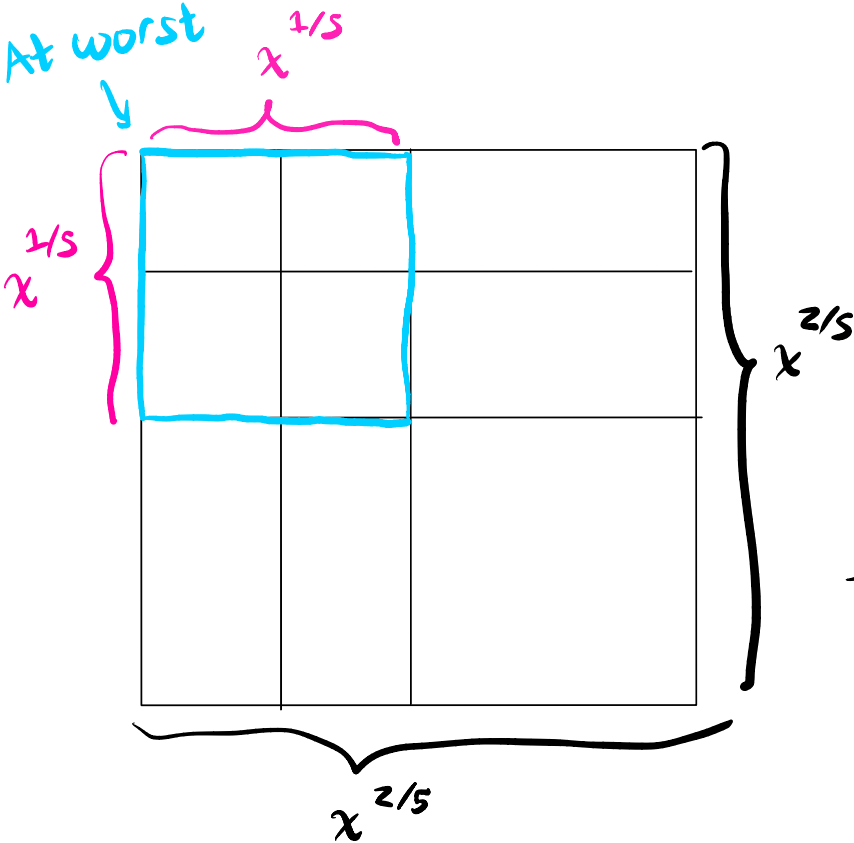
* In the case where m or n is in a "bad" residue class $(\mathrm{mod}\ q)$, we show that $L_2 - L_1$, $L_1 - L_0$ are char. functions of intervals (or unions of intervals).

* So, we just need to compute a table of

$$\sum_{\substack{m \equiv a_{\text{bad}} \pmod q \\ m \in I_{\text{bad}}}} \mu(m)$$

which requires pre-computing a table of values of $\mu(m)$; can be done in time $O(b)$ & space $O(b \log b)$

Savings: a factor of "a" compared with the naive method



naive

$$\text{Time}: \approx \frac{x^{4/5}}{x^{1/5}} = \boxed{x^{3/5}}$$

savings

Computations

We wrote our algorithm in C++ and ran it on an 80-core machine at the Max Planck Institute for Mathematics:

x	$M(x)$
10^{17}	-21830254
10^{18}	-46758740
10^{19}	899990187
10^{20}	461113106
10^{21}	-3395895277
10^{22}	-2061910120
10^{23}	62467771689

Kuznetsov

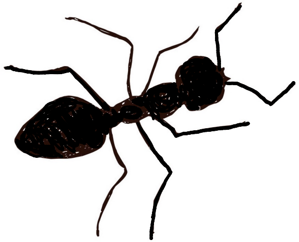
x	$M(x)$
2^{68}	2092394726
2^{69}	-3748189801
2^{70}	9853266869
2^{71}	-12658250658
2^{72}	9558471405
2^{73}	-6524408924
2^{74}	-6336351930
2^{75}	-4000846218

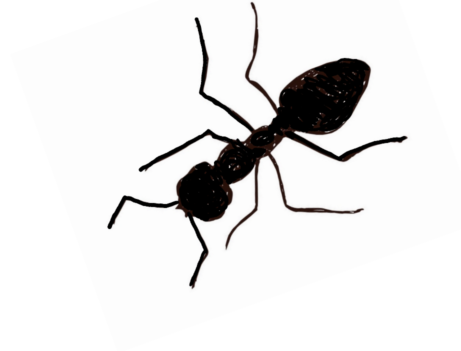
Hurst

H-T '21

These match with Kuznetsov & Hurst's Computations using Deleglise & Rivat's algorithm

except for a possible sign error at $x=10^{21}$





Thank you!