Abstract

we introduce an algorithm that generates primes included in a given interval \( I = [a, b] \), the algorithm is an optimization to the segmented sieve of eratosthenes, it finds primes up to \( N \) without any repetition of multiples of primes using the equation \( p^2_n.p_j + 2p_n.p_j.c = N \) with \( c \in \mathbb{Z}^+ \), its time complexity is sublinear \( O(n \log \log(n) - n(\log \log(n))^2) \).

1 Introduction

A prime number (or a prime) is a natural number greater than 1 that has no positive divisors other than 1 and itself. A composite number is a positive integer that has at least one positive divisor other than one or the number itself. In other words, a composite number is any integer greater than one that is not a prime number. Every composite number can be written as the product of two or more (not necessarily distinct) primes, for example, the composite number 299 can be written as 13 \times 23, and that the composite number 360 can be written as 23 \times 32 \times 5; furthermore, this representation is unique up to the order of the factors. This is called the fundamental theorem of arithmetic.

A prime sieve or prime number sieve is a fast type of algorithm for finding primes. There are many prime sieves. The simple sieve of Eratosthenes, the sieve of Sundaram, the still faster but more complicated sieve of Atkin, and various wheel sieves, are most common. A prime sieve works by creating a list of all integers up to a desired limit and progressively removing composite numbers (which it directly generates) until only primes are left. This is the most efficient way to obtain a large range of primes. In this present research we produce some equations and do some modifications to avoid the repetition of multiples of primes to optimize the segmented sieve of eratosthenes by reducing the number of operations to only \( O(N^{2/3}) \), a special rarely if ever implemented segmented version of the sieve of Eratosthenes, with basic optimizations, uses \( O(N) \) operations and \( O(n^{1/2} \log \log n / \log n) \) bits of memory.

2 The relation between primes and odd composite numbers

To minimize the number of operations of the new algorithm, only the odd composite numbers are crossed off without repetition and by consequence the odd numbers left must be primes. For this, one needs to find a relationship between primes and odd composite numbers.

Let’s put all the odd composite included in the interval \( I = [0, 100] \).

\[
9 = 3 \times 3 \\
15 = 3 \times 5 \\
21 = 3 \times 7 \\
27 = 3 \times 9 
\]
33 = 3 \times 11 \\
39 = 3 \times 13 \\
45 = 3 \times 15 \\
51 = 3 \times 17 \\
57 = 3 \times 19 \\
63 = 3 \times 21 \\
69 = 3 \times 23 \\
75 = 3 \times 25 \\
81 = 3 \times 27 \\
87 = 3 \times 29 \\
91 = 3 \times 31 \\
99 = 3 \times 33 \\

odd composites above are of the form \( N = 3 \times (3 + (2 \times n)) \)

25 = 5 \times 5 \\
35 = 5 \times 7 \\
55 = 5 \times 11 \\
65 = 5 \times 13 \\
85 = 5 \times 17 \\
95 = 5 \times 19 \\

odd composites above are of the form \( N = 5 \times (5 + (2 \times n)) \)

49 = 7 \times 7 \\
77 = 7 \times 11 \\
91 = 7 \times 13 \\

odd composites above are of the form \( N = 7 \times (7 + (2 \times n)) \)

2.1 Explanation

for any given odd composite \( N \), this can be written as

\[ p_n^2 + 2p nc = N \]  

(1)

with \( c \in \mathbb{Z}^+ \) and \( p_n \) are all the primes except 2 which satisfy \( p_n \leq \sqrt{N} \), the equation 1 is true since if we devide it by \( p_n \) we get the trivial equation for odd numbers. For a given interval \( I = [a, b] \) one calculates the constant \( c \) and iterates to generates the odd composites included in the interval \( I \). \( a \leq p_n^2 + 2p nc \leq b \). This is similar to the Eratosthenes’s algorithm [6] as we start crossing off multiples of \( p \) at \( p^2 \). We know that more than a third of our composites are divisible by 3, and more than a fifth of them are divisible by 5 but how many times the same composites are generated by different prime multiples? (e.g. \( 3 \times 15 = 45 \) and \( 5 \times 9 = 45 \)) can we find an efficient way to avoid redundant eliminations from the prime list so that we can reduce the number of operations to generate composite? This is just the key of this new algorithm.

2.2 The redundant multiples of primes

The main idea of this new algorithm is to avoid crossing off the redundant multiples of primes. this way a composite is only crossed off once, indeed using the equation 1, for a different two primes \( p_n \) and \( p_j \) with \( p_n < p_j \) with the same multiples we get:

\[ p_n^2 + 2p_n c_1 = p_j^2 + 2p_j c_2 = N \]  

(2)

Pratically the equation 2 will not give us too much interest as we have 2 unkown \( c_1 \) and \( c_2 \) to be determined. A better choice to do is multiplying the equation 1 by \( p_j \) to
have only one unknown constant $k$. Thus

$$p_n^2.p_j + 2p_n.p_j.k = N \quad (3)$$

Below is a table of some equations of the same multiples of 3 and another primes.

<table>
<thead>
<tr>
<th>$p_n$</th>
<th>$p_j$</th>
<th>$p_n^2.p_j + 2p_n.p_j.k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>5</td>
<td>45+30k</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>63+42k</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>99+66k</td>
</tr>
<tr>
<td>3</td>
<td>13</td>
<td>117+78k</td>
</tr>
<tr>
<td>3</td>
<td>17</td>
<td>153+102k</td>
</tr>
<tr>
<td>3</td>
<td>19</td>
<td>171+114k</td>
</tr>
</tbody>
</table>

using the equation 3 when implementing one can avoid all the duplicated multiples of primes included in a given interval $I = [a,b]$.

$$a \leq p_n^2.p_j + 2p_n.p_j.k \leq b \quad (4)$$

or in term of $k$

$$\frac{a - p_n^2.p_j}{2p_n.p_j} \leq k \leq \frac{b - p_n^2.p_j}{2p_n.p_j} \quad (5)$$

So by using the equation 1 and 5 for finding composites in a given interval $I$, one increments by $c$ to cross off all multiples of 3, then for finding multiples of 5, one increments by $c$ except the values given by $k$ to not to cross of the same multiples of 3, the same thing for multiples of 7, one increments by $c$ given by the equation 1 except the values of $c$ given by $k_1$ (the same multiples of 3) and $k_2$ (the same multiples of 5) given by the equation 5 and so on ..., we can calculate the relation between $c$ and $k$ as follows:

$$p_j^2 + 2p_j.c = p_n^2.p_j + 2p_n.p_j.k \quad (6)$$

Then by rearranging in term of $c$:

$$c = \frac{p_n^2 - p_j}{2} + p_n.k \quad (7)$$

then when looping by $c$ using the equation 1, we skip the multiples crossed off earlier by using the equation 7 for the values of $k$ given by the equation 5 eg: for a given interval $[a,b]$ if $a = 50$ and $b = 120$, first of all we use the equation 1 to cross off multiples of 3, we get (51,57,63,69,75,81,87,93,99,105,111,117) and to cross off multiples of the next prime 5 by skipping multiples divisible by 3 crossed off earlier we use equation 1, equation 5 and equation 7 for $p = 5$, equation 1 gives $3 \leq c \leq 9$, equation 5 gives for $p_n = 3$ and $p_j = 5 : 1 \leq k \leq 2$ and equation 7 gives $c = 2 + 3.k$, then the iterations to not to do for $c$ are $c = (5,8)$. this gives the multiples of 5 (55,65,85,95,105), we repeat the same process for multiples divisible by the next prime 7. we use the equation 1 to find values of $c$ to iterate and we use equation 5 and equation 7 to skip the values of $c$ where the multiples were already crossed off by 3 and 5. for $N = b - a$, this skipping reduces the number of operations by $\frac{N}{2}$. Moreover in practice one should know when and how the redundant multiples occur, of course not all the primes $p_n \leq \sqrt{b}$ generate the same multiples in the range $[a,b]$, it could be out of that range, eg: if $b = 196$, the primes from 3 to $\sqrt{196}$ are: 3, 7, 11, 13, if we use directly the equations 5 and 7 to avoid the multiples of say 7 and 11, this gives $7^2 \times 11 = 539$ which is higher than $b$, so for a better use of these equations one must consider the condition $p_n^2.p_j \leq b$, by putting $p_n = p_j - \alpha$ where the $\alpha$ denotes the gap between $p_n$ and $p_j$, this gives

$$\alpha \geq p_j - \sqrt{\frac{b}{p_j}} \quad (8)$$
so for \( b = 196 \) and \( p_j = 11 \), \( \alpha \geq 7 \), pratically meaning ,we should only consider the redundant multiples crossed off by the primes inferior or equal to \( p_j - \alpha \), in this example it is only with primes 11 and 3. the equation 8 permits to avoid the useless iterations and therefore it reduces the number of operations.as the \( \alpha \) denotes the prime gap it should not be negative and as its minimum value is 2,that means \( p_j - \sqrt{\frac{p_j}{2}} \) must be higher or equal to 2 , theoretically it is a cubic function to solve \( b \leq p_j^3 - 2p_j^2 + 4p_j \),in practice to reduce the time complexity we consider only that \( \alpha \geq 0 \),which gives \( p_j \geq \sqrt[3]{b} \). its meaning is for all of the primes \( \geq \sqrt[3]{b} \),one skips to calculate the redundant multiples of \( p_j \) divisible by the precedents \( p_n \) inferior or equal to \( p_j - \alpha \).And for all of the primes \( \leq \sqrt[3]{b} \) one avoids to cross off all the redundant multiples of \( p_j \) divisible by all the primes inferior to it.

3 The algorithm

As the segmented sieve of Eratosthenes,for a given interval \([a, b]\), we start by calculating prime numbers from 3 up to \( \sqrt{b} \),then after having found them and stored them we start sieving in the interval \([a, b]\) in two big steps.each step is divided into two substeps, the algorithm concept is introduced as follow:

1. Prime sieving up to \( \sqrt{b} \)

To find primes in the interval \([a, b]\), we must first find the primes from 3 up to \( \sqrt{b} \),for this ,the first thing to do is to use equation 1 to cross off all multiples of 3 up to \( \sqrt{b} \),then for the next multiples is explained in the next steps

(a) Step 1.1 for \( p_j \leq \sqrt{b} \)

After having crossed off all multiples divisible by 3 , we go to the next uncrossed number (wich is prime) up to \( p_j \leq \sqrt[3]{b} \) \( (\sqrt[3]{b} = \sqrt[3]{\sqrt[3]{b}} \) as we’re working up to \( \sqrt{b} \) explained in paragraph (2.2) ),by using equation 1 we iterate \( c \) except its values given by \( k \) in the equation \( 7 \),\( (k \) goes from zero to \( \sqrt[3]{p_j - p_i} \)) to avoid the redundant multiples divisible by the earliar prime \( 3 \),after that we move to the next uncrossed number wich is prime 5,we use the same equation as before to iterate \( c \) to cross off multiples divisible by 5 and to avoid crossing of multiples divisible by the earliar prime \( 3 \) we continue the same procces up to the prime \( p_j \leq \sqrt[3]{b} \) by iterating \( c \) from zero ( starting from \( p_j^3 \)) to \( c = \sqrt[3]{\sqrt[3]{p_j - p_i}} \) and avoiding the values of \( c \) given by equation 7 for all the primes inferior to it.

(b) Step 1.2 for \( p_j > \sqrt[3]{b} \)

In this step we work only with the numbers \( p_j > \sqrt[3]{b} \), in fact we use the same method in the step (1.1) except in this step we calculate the \( \alpha \) given by equation 8 for each prime found \( \alpha \geq p_j - \sqrt[3]{\sqrt[3]{p_j}} \) then we iterate \( c \) from zero to \( c = \sqrt[3]{\sqrt[3]{p_j - p_i}} \) ,except its values given by equation 7 ,but the difference here we don’t calculate values of \( k \) of all the primes inferior to the prime found ,but just the primes \( p_n \leq p_j - \alpha \).we continue sieving up to \( \sqrt[3]{b} = \sqrt[3]{\sqrt[3]{b}} \).arriving at this point we’ve crossed off all the composites \( \leq \sqrt[3]{b} \) the odd numbers left are prime, we store them for finding their multiples in the interval \([a, b]\).

2. Prime sieving in the interval \([a, b]\)

After having found and stored primes in step(1) ,we use them to find primes included in the given interval \([a, b]\) by eliminating all composites in this interval.the step (2) is similar to the precedent one except for the minimum values of \( c \) and \( k \).Depending on the value of the minimum range \( a \), the values of these
constants could be negatives for higher primes, for this purpose and to avoid verifications at each prime or at each iteration, we divide this second step into 2 big steps, the first step we deal with the primes \( p_j < \sqrt{3} \), as detailed above we use the values of \( k \) for all primes from 3 to the prime \( p_n \) inferior to the prime just found \( p_j \), in the second step of the algorithm we deal with the \( p_j > \sqrt{3} \) where we use the values of \( k \) for primes from 3 to \( p_n \leq p_j - \alpha \), where the \( \alpha \) is calculated by 8. The signs below calculated by 8.

(a) Step 2.1 for \( p_j \leq \sqrt{3} \)

i. Step 2.1.1 for \( p_n \leq \sqrt{a} \)

A. Step 2.1.1.1 for \( \alpha \leq p_j < \sqrt{a} \)

the minimum values of \( k \) and \( c \) are \( \leq 0 \), we use equation 1 to iterate \( c \) from 0 to \( c = \left\lfloor \frac{b-p_j^2}{2p_j} \right\rfloor \), by skipping its values (given by \( k \) in equation 7) from 0 to \( k = \left\lfloor \frac{b-p_j^2}{2p_n-p_j} \right\rfloor \), \( (p_n \) are all the primes from 3 to the prime inferior to the prime just found \( p_j \)).

B. Step 2.1.1.2 for \( \alpha > p_j \leq \sqrt{a} \)

the minimum values of \( c \) are negative but not for \( k \), in this case we iterate \( c \) from 0 to \( c = \left\lceil \frac{b-p_j^2}{2p_j} \right\rceil \), by skipping its values given by \( k \) from \( k = \left\lceil \frac{a-p_j^2}{2p_n-p_j} \right\rceil \) to \( k = \left\lceil \frac{b-p_j^2}{2p_n-p_j} \right\rceil \).

ii. Step 2.1.2 for \( p_n > \sqrt{a} \)

A. Step 2.1.2.1 for \( \alpha \leq p_j < \sqrt{a} \)

the minimum values of \( k \) and \( c \) are negative but not for \( c \), we iterate \( c \) from \( c = \left\lfloor \frac{a-p_j^2}{2p_j} \right\rfloor \) to \( c = \left\lceil \frac{b-p_j^2}{2p_j} \right\rceil \), by skipping its values given by \( k \) from 0 to \( k = \left\lceil \frac{a-p_j^2}{2p_n-p_j} \right\rceil \).

B. Step 2.1.2.2 for \( \alpha > p_j \leq \sqrt{a} \)

the minimum values of \( k \) and \( c \) are positive, we iterate \( c \) from \( c = \left\lfloor \frac{a-p_j^2}{2p_j} \right\rfloor \) to \( c = \left\lceil \frac{b-p_j^2}{2p_j} \right\rceil \), by skipping its values given by \( k \) from \( k = \left\lfloor \frac{a-p_j^2}{2p_n-p_j} \right\rfloor \) to \( k = \left\lceil \frac{b-p_j^2}{2p_n-p_j} \right\rceil \).

(b) Step 2.2 for \( p_j > \sqrt{3} \)

i. Step 2.2.1 for \( p_n \leq \sqrt{a} \)

A. Step 2.2.1.1 for \( \alpha \leq p_j \leq \sqrt{a} \)

the minimum values of \( k \) and \( c \) are \( \leq 0 \), we iterate \( c \) from 0 to \( c = \left\lfloor \frac{b-p_j^2}{2p_j} \right\rfloor \), by skipping its values given by \( k \) from 0 to \( k = \left\lfloor \frac{b-p_j^2}{2p_n-p_j} \right\rfloor \), \( (p_n \) are all the primes from 3 to \( p_n \leq p_j - 2 \) given by equation 8.

B. Step 2.2.1.2 for \( \alpha > p_j > \sqrt{a} \)

the minimum values of \( k \) are positives and \( c \) are \( \leq 0 \), we iterate \( c \) from 0 to \( c = \left\lceil \frac{b-p_j^2}{2p_j} \right\rceil \), by skipping its values given by \( k \) from \( k = \left\lfloor \frac{a-p_j^2}{2p_n-p_j} \right\rfloor \) to \( k = \left\lceil \frac{b-p_j^2}{2p_n-p_j} \right\rceil \).

ii. Step 2.2.2 for \( p_n > \sqrt{a} \)

A. Step 2.2.2.1 for \( \alpha \leq p_j \leq \sqrt{a} \)

the minimum values of \( k \) are \( \leq 0 \) and of \( c \) are positives, we iterate
It’s well known that where

The Euler-Maclaurin Summation Formula says:

c from \(c = \left[ \frac{a-p^2}{2p} \right]\) to \(c = \left[ \frac{b-p^2}{2p} \right]\), by skipping its values given by \(k\) from 0 to \(k = \left[ \frac{b-p^2}{2p_n-p_j} \right]\).

B. Step 2.2.2.2 for \(a > p_j - \sqrt{\frac{p}{p_j}}\)

the minimum values of \(k\) and \(c\) are both positives, we iterate \(c\) from \(c = \left[ \frac{a-p^2}{2p} \right]\) to \(c = \left[ \frac{b-p^2}{2p} \right]\), by skipping its values given by \(k\) from \(k = \left[ \frac{a-p^2}{2p_n-p_j} \right]\) to \(k = \left[ \frac{b-p^2}{2p_n-p_j} \right]\).

4 Computational analysis

For large \(b\) let’s put \(b - a \approx N\) and \(p_1 = 2; p_2 = 3; p_3 = 5; p_4 = 7\). . . The number of iterations the algorithm does is:

\[
S_N = \frac{N}{2} \sum_{n=3}^{p_n \leq \sqrt{N}} \left( \frac{1}{p_n} + 1 \right) - \frac{N}{2} \sum_{n=3}^{p_n \leq \sqrt{N}} \sum_{p_j=5}^{p_j \leq \sqrt{N}} \left( \frac{1}{p_n p_j} + 1 \right)
\]

since we have \(p_n \approx n \log(n)\) then \(\sum_{n=2}^{\sqrt{N}} \frac{1}{p_n} \approx \log\log N\). so the sum \(\sum_{n=3}^{p_n \leq \sqrt{N}} \sum_{p_j=5}^{p_j \leq \sqrt{N}} \left( \frac{1}{p_n p_j} \right)\)
grows like \(\sum_{n=3}^{\sqrt{N}} \frac{\log\log(n)}{n \log(n)}\)

The Euler-Maclaurin Summation Formula says:

\[
\sum_{k=5}^{m} \frac{\log(\log(k))}{k \log(k)} = \frac{1}{2} \log(\log(m))^2 + C + O\left(\frac{\log(\log(m))}{m \log(m)}\right)
\]

Therefore,

\[
\sum_{k=5}^{\sqrt{N}} \frac{\log(\log(k))}{k \log(k)} = \frac{1}{2} \log(\sqrt{\pi})^2 + C + O\left(\frac{\log(\log(n))}{\sqrt{n} \log(n)}\right)
\]

\[
= \frac{1}{2} \log\left(\frac{1}{2} \log(n)\right)^2 + C + O\left(\frac{\log(\log(n))}{\sqrt{n} \log(n)}\right)
\]

\[
= \frac{1}{2} \log(\log(n))^2 - \log(2) \log(\log(n)) + \frac{1}{2} \log(2)^2 + C + O\left(\frac{\log(\log(n))}{\sqrt{n} \log(n)}\right)
\]

where \(C \doteq -0.08334404437765197472024727705275296252855\).

It’s well known that \(\sum_{n=3}^{\sqrt{N}} \frac{1}{p_n} \approx N \log\log N\), then by neglecting the error terms, the total number of iteration of the algorithm is:

\[
S_N \approx n \log\log(n) - n(\log\log(n))^2
\]

References


