Prime Generating Algorithms by Skipping Composite Divisors

Neeraj Anant Pande
Assistant Professor
Department of Mathematics & Statistics
Yeshwant Mahavidyalaya (College), Nanded-431602
Maharashtra, INDIA
napande@gmail.com

Abstract — Three elementary versions of simple prime generating sieves have already been improved by skipping even divisors other than 2. All composite integers are multiples of primes. Taking help of the transitivity property of divisibility allows using the logic that if a prime doesn't divide a number, then any composite number which is multiple of that prime also cannot divide it. That altogether eliminates the necessity of trying composite numbers for divisibility in primality tests and gives the next generation of prime generating sieves. In fact, the best version of this generation happens to be the celebrated and historic Sieve of Eratosthenes.

Keywords - Algorithm; Sieve; Prime Number

Mathematics Subject Classification 2010: 11Y11, 11Y16, 65Y04, 65Y20, 68Q25

I. INTRODUCTION

The very definition of prime numbers depends on divisibility [2] and [4]. Using the fundamental properties of divisibility, three elementary sieves are devised for prime number generation which are also exhaustively discussed and compared in [5]. The refinement of all sieves in [5] to the corresponding ones in [6] is by using a fundamental property that if the number 2 doesn't divide any positive integer, then any even integer also cannot divide it. This eradicates the unnecessary checking of other even numbers as possible divisors in the process of determining primality and adds significant efficiency in the process.

This is very true that in prime number study, number 2 enjoys unique status due to many specialties like being first prime, being the only even prime etc. But the property used in refining algorithms in [5] to those in [6] is not limited to 2. This is why all sieves in [6] are also prone to further improvement on the same line and this work presents the same.

II. REFINED SIEVE OF SUBTYPE 1

Taking hint from the sieve numbering in [6], for consistent reference purpose, here we renumber the Sieves 1, 2 and 3 of [5] as Sieves 1.1.1, 1.1.2 and 1.1.3, respectively. Also we adopt the convention of identifying the sieve subtypes by the very last digit used in denoting them.

The property of prime numbers is that they don't have positive integral divisors other than 1 and themselves. Anyway, not only for primes but for all positive integers *k*, the positive integral divisors cannot exceed *k*. So sieves subtypes 1 have selected the range of numbers 2 to $k - 1$ for possible occurrence of divisors. This is very elementary approach. The renumbered Sieve 1.1.1 from [5] is based on the same :

```
Take an integer $n$ larger than 1
For all values of $k$ from 2 to $n$
    For values of integer $d$ from 2 to $k-1$
        If $d$ divides $k$ perfectly,
            Stop checking as $k$ is not prime
        Else
            Continue checking with next value of $d$
    If checks do not stop for any value of $d$ till $k-1$, $k$ is prime
    Take next value of $k$
```

The number 2 is clearly prime. All even numbers are divisible by 2. So, using the transitive property of divisibility, for any larger number divisible by any even divisor, it would be divisible by 2. The other way, if it is not divisible by 2, it cannot be divisible by any even divisor. This logic eliminates the need for testing even numbers greater than 2 as divisors, which has led to Sieve 2.1.1 of [6] :

```
Take an integer $n$ larger than 1
2 is prime
```

```
For all values of $k$ from 3 to $n$
    For values of integer $d$ from 2 to $k-1$ and only odd values after 2
        If $d$ divides $k$ perfectly,
            Stop checking as $k$ is not prime
        Else
            Continue checking with next value of $d$
    If checks do not stop for any value of $d$ till $k-1$, $k$ is prime
    Take next value of $k$
```

The property of the number 2 used in improving Sieve 1.1.1 to Sieve 2.1.1 is in fact no way unique to 2 only. For any positive integer *d*, it is equally true that if *d* divides a number *m* and *m* divides our integer *k*, by transitivity of divisibility, *d* divides *k*. Contrapositively, if *k* is not divisible by some *d*, it cannot be divisible by any multiple of *d*, eliminating the necessity of testing multiples of *d* as divisors. Summarily, only smaller prime numbers are enough to be tested as divisors in primality tests. This give us new Sieve 3.1.1 :

```
Take an integer $n$ larger than 1
2 is prime
For all values of $k$ from 3 to $n$
    For values of integer $d$ from 2 to $k-1$ and only prime values
        If $d$ divides $k$ perfectly,
            Stop checking as $k$ is not prime
        Else
            Continue checking with next value of $d$
    If checks do not stop for any value of $d$ till $k-1$, $k$ is prime
    Take next value of $k$
```

The reduction in the number of tests required is so significant making this a superior improvement for sieves of subtypes 1. Since sieves of [5] are very inefficient, it has been necessary to restrict the range of numbers to 1 – 100000 only. Following comparison is self-explanatory.

TABLE 1: NUMBER OF STEPS TAKEN BY SIEVES OF SUBTYPE 1

Sr. No.	Range	Number of Primes Found	Steps Taken by Sieve 1.1.1	Steps Taken by Sieve 2.1.1	Steps Taken by Sieve 3.1.1
1	1 – 10	4	15	12	12
2	1 – 100	25	1133	628	411
3	1 – 1000	168	78022	39676	15620
4	1 – 10000	1229	5775223	2894496	776631
5	1 – 100000	9592	455189150	227664778	46314477

These results can be graphically compared for quick illustration.

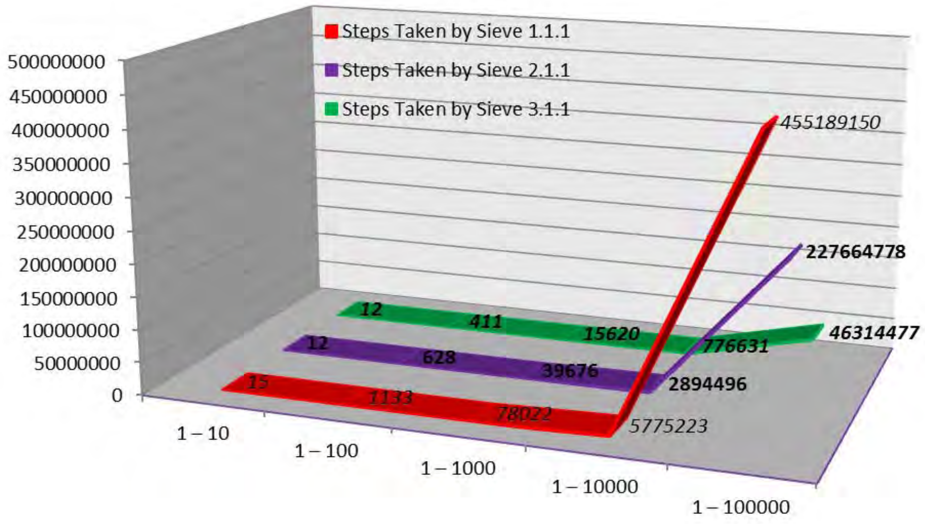

FIGURE 1 : COMPARISON OF STEP REQUIREMENTS OF SIEVES OF SUBTYPE 1

III. REFINED SIEVE OF SUBTYPE 2

As Sieves 1.1.1 and 2.1.1 have been modified to Sieve 3.1.1 by considering only prime divisors in deciding primality, their corresponding improved versions Sieves 1.1.2 in [5] and 2.1.2 in [6] can be modified to new Sieve 3.1.2 adopting the same approach :

```
Take an integer $n$ larger than 1
2 is prime
For all values of $k$ from 3 to $n$
    For values of integer $d$ from 2 to $k/2$ and only prime values
        If $d$ divides $k$ perfectly,
            Stop checking as $k$ is not prime
        Else
            Continue checking with next value of $d$
    If checks do not stop for any value of $d$ till $k/2$, $k$ is prime
    Take next value of $k$
```

The reduction in the range of divisors to half contributes to further efficiency in all previous successor procedures.

TABLE 2: NUMBER OF STEPS TAKEN BY SIEVES OF SUBTYPE 2

Sr. No.	Range	Number of Primes Found	Steps Taken by Sieve 1.1.2	Steps Taken by Sieve 2.1.2	Steps Taken by Sieve 3.1.2
1	1 – 10	4	9	10	9
2	1 – 100	25	616	376	293
3	1 – 1000	168	40043	20730	9574
4	1 – 10000	1229	2907640	1461014	436328
5	1 – 100000	9592	227995678	114070446	25042950

Graphical depiction of the numbers follows :

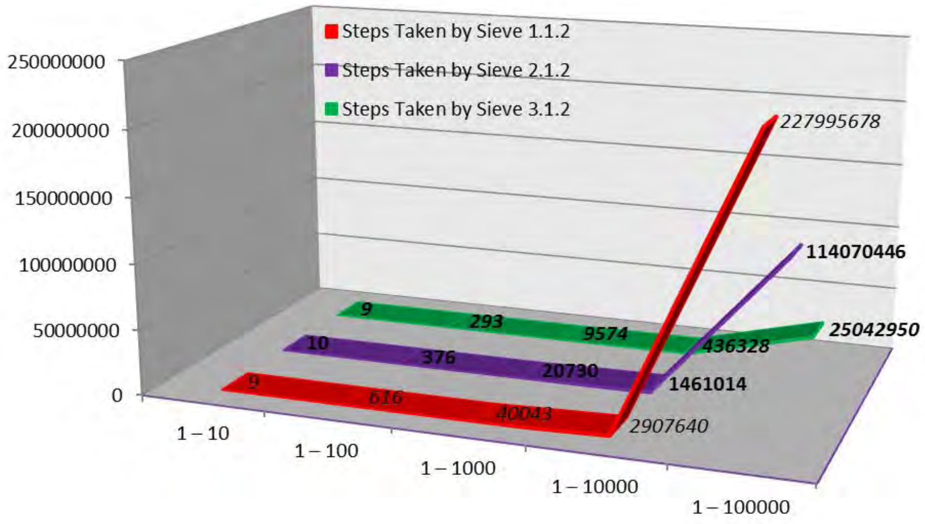

FIGURE 2 : COMPARISON OF STEP REQUIREMENTS OF SIEVES OF SUBTYPE 2

IV. REFINED SIEVE OF SUBTYPE 3

Sieves 1.1.3 in [5] and 2.1.3 in [6] have been modified from Sieves 1.1.1, 1.1.2 and 2.1.1, 2.1.2 respectively by restricting the range of divisors to square root of the number being tested for primality. Applying this method to new Sieves 3.1.1 & 3.1.2 yields 3.1.3 :

```
Take an integer $n$ larger than 1
2 is prime
For all values of $k$ from 3 to $n$
    For values of integer $d$ from 2 to $\sqrt{k}$ and only prime values
        If $d$ divides $k$ perfectly,
            Stop checking as $k$ is not prime
        Else
            Continue checking with next value of $d$
    If checks do not stop for any value of $d$ till $\sqrt{k}$ , $k$ is prime
    Take next value of $k$
```

The subtypes 3 have always been the best versions than 1 and 2, and this tradition continues; with of course even earlier subtypes 3 being improved by present Sieve 3.1.3.

TABLE 3: NUMBER OF STEPS TAKEN BY SIEVES OF SUBTYPE 2

Sr. No.	Range	Number of Primes Found	Steps Taken by Sieve 1.1.3	Steps Taken by Sieve 2.1.3	Steps Taken by Sieve 3.1.3
1	1 – 10	4	8	9	8
2	1 – 100	25	236	185	181
3	1 – 1000	168	5288	3349	2801
4	1 – 10000	1229	117527	65956	43753
5	1 – 100000	9592	2745694	1445440	744436

These figures have the following say diagrammatically :

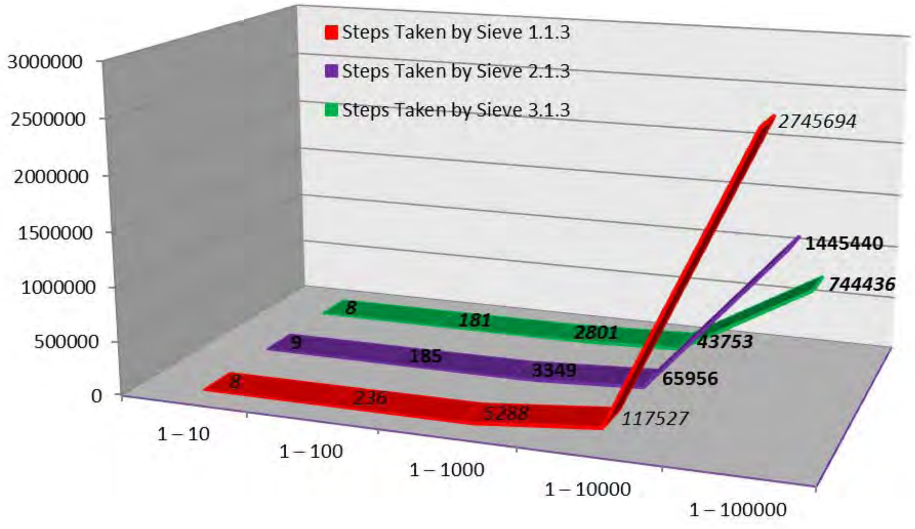

FIGURE 3 : COMPARISON OF STEP REQUIREMENTS OF SIEVES OF SUBTYPE 3

The sieves in the earlier works [5] and [6] have laid the foundation for the three new versions of the sieves presented here, viz., Sieves 3.1.1, 3.1.2 and 3.1.3.

It is immediately noted that Sieve 3.1.3 is nothing else but the most celebrated ancient Sieve of Eratosthenes! It has evolved here in the process of successive refinements of earlier elementary sieves. It is really appreciable that the sieve which evolves gradually after devising 11 sieves was invented by Eratosthenes of Cyrene as back as in 200 BC. Tributes to the genius of the great philosopher who could identify the right efficient version of this succession of sieves without the explicit gradual foundation of what we have.

There is an overhead involved in these versions of sieves in the form of the list of prime numbers that one has to have while doing divisibility tests. So the methods presented here cannot be appropriate for determining primality of a random number directly without having a sufficient list of prime numbers to work out the procedures. But if their algorithms are run for ranges starting with 2, automatically they themselves generate the all required successive primes which can be stored for subsequent test purposes.

A comparison of the three newly presented sieves also reflects the superiority of the third one, viz., 3.1.3, the Sieve of Eratosthenes.

TABLE 4: NUMBER OF STEPS TAKEN BY SIEVES SKIPPING COMPOSITE DIVISORS

Sr. No.	Range	Number of Primes Found	Steps Taken by Sieve 3.1.1	Steps Taken by Sieve 3.1.2	Steps Taken by Sieve 3.1.3
1	1 – 10	4	12	9	8
2	1 – 100	25	411	293	181
3	1 – 1000	168	15620	9574	2801
4	1 – 10000	1229	776631	436328	43753
5	1 – 100000	9592	46314477	25042950	744436

The graphical inter-se comparison of new sieves is as follows :

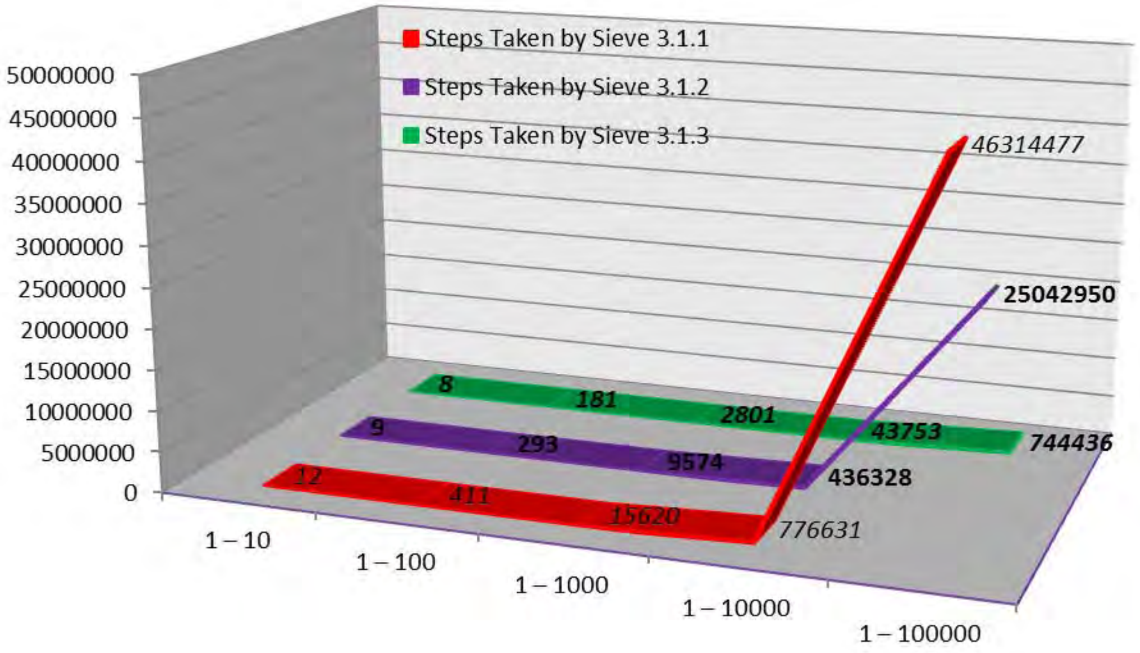

FIGURE 4 : INTER-SE COMPARISON OF STEP REQUIREMENTS OF NEW GENERATION SIEVES

ACKNOWLEDGEMENTS

The author is grateful to the Java (7 Update 25) Programming Language Development Team and the NetBeans IDE 7.3.1 Development Team, for the free use of their software in implementing the algorithms.

The author thanks the University Grants Commission (U.G.C.), New Delhi of the Government of India for funding this research work under a Research Project (F.No. 47-748/13(WRO)).

REFERENCES

[1] Isaac Asimov, “Asimov's Biographical Encyclopedia of Science and Technology”, (New Revised Edition) Pan Books Ltd, London, 1975.

[2] David M. Burton, “Elementary Number Theory”, Tata McGraw-Hill Education, 2007.

[3] Donald E. Knuth, “The Art of Computer Programming, Volume 1: Fundamental Algorithms”, Addison-Wesley, Reading, MA, 1968.

[4] Evan Niven, Herbert S. Zuckerman, Huge L. Montgomery, “An Introduction to the Theory of Numbers”, John Wiley & Sons Inc., U.K., 2008.

[5] Neeraj Anant Pande, “Evolution of Algorithms: A Case Study of Three Prime Generating Sieves”, Journal of Science and Arts,13-3(24), 2013, pp. 267-276.

[6] Neeraj Anant Pande, “Algorithms of Three Prime Generating Sieves Improvised by Skipping Even Divisors (Except 2)”, American International Journal of Research in Formal, Applied and Natural Sciences. 4(1), 2013, pp. 22 - 27.

[7] Herbert Schildt , “Java : The Complete Reference” (7th Edition), Tata McGraw - Hill Education, 2006.