International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 8, pp: 337-343www.ijaem.net ISSN: 2395-5252

Farming of Primes:An algorithm to cultivate primes from first known N-primes

Arpit U. Thool[1] and Uday S. Thool[2*]

[1] *Associate Engineer, Acquia India Private Limited, 4th floor Office No. 4, Cerebrum B3, Klyani Nagar Pune, Maharashtra State, INDIA.* [2] *Associate Professor, Department of Mathematics, Institute of Science Nagpur, 440001, Maharashtra State, INDIA.*

**Corresponding author: Uday S. Thool*

Date of Submission: 10-10-2020 Date of Acceptance: 30-10-2020

ABSTRACT: Prime numbers plays a major role in number theory and cryptography. This paper presents a new algorithm to generate primes without performing any divisions. It is simple, efficient, fast and different from the sieve of Eratosthenes. In addition to this, the work also introducesnew sieve method to extract all primes less than or equal to $2N + 1$, which is significantly faster than the sieve of Eratosthenes method. To verify whether the number is prime or not, it is sufficient to test its divisibility by all primes less than or equal to the square root of that number. Here we have used the same principle to generate primes called farming of primes. For this, we have used first N primes as our seeds and generated all primes lessthan the square of (1+ N^{th} prime or $2 + N^{th}$ prime) accordingly, whether $N = 1$ or$N > 1$. That is, we can generate a few extra primes by making use of first N primes. It guarantees at least 4-times of yield! For N = 1, we get a total of 4 primes smaller than $(1 + 2)^2 = 9$. For $N = 2$, we will have all nine primes smaller than 25. For $N = 303$, we will have 283413 primes. Because of this, we have named this algorithm as the farming of primes. We have also developed a simple computer program for its implementations.

Key words:Algorithm of Primes, Mersenne primes, the sieve of Eratosthenes, Farming of primes, Python program.

I. INTRODUCTION:

There is always a quest for finding a newer, faster method for the generation of primes. Ancient Greek Mathematicians first extensively studied the Prime numbers and their properties. In 250 BC, the Greek Eratosthenes devised an algorithm for extracting primes from the bed of first N natural numbers with tremendous speed [9]. The Mersenne prime numbers $M_n = 2^N - 1$ are of great importance. According to Great Internet Mersenne Prime Search (GIMPS) by the year 2018, a total of 50 Mersenne primes are known. Rivest-Shamir-Adleman (RSA) algorithm is a cryptographic algorithm, and primes play a vital role in encryption/decryption of messages and the size of prime numbers used, dictate how secure the encryption will be [1, 2, 18]. The product formed by using two large prime numbers makes the code much harder to crack. In number theory, cryptography, signal processing, and computational Mathematics primes play a significant role.

Nevertheless, testing of primality of large numbers [16] is a challenging, time-consuming, and cumbersome job [5]. In the computer domain, the finding of primes is significant. Noticeably, there are several theories on prime, but the cost of computation of big prime numbers is enormous. There is no known formula to generate all the primes, though there are several algorithms to generate primes[3, 4, 5, 15]**.** Thool and Thul 2002, conjectured that, if $P_0 = 2$, then $P_{n+1} = 2^{P_n} - 1$is always prime for all $n = 0, 1, 2, \ldots$ and hence Mersenne primes are infinite [19].

Surprisingly prime number generation algorithms are extensively worked on and many of the researchers claim for the efficient generation of primes [6, 10, 11, 13, 14]. This paper introduces a new technique that, if $P_1, P_2, \ldots, P_N$ are the first N known primes, then we can generate all primes less than either 3^2 or $(P_N + 2)^2$according to the value of $P_N$is two or greater than two without performing any divisions.

II. PRELIMINARIES:

Definition: An integer greater than one is calledprime number if its only positive divisors are one and itself.

Mathematically, an integer $p > 1$ is prime if $q|p$ for some integer$q > 0$, then either $q = 1$ or $q = p$.Equivalently integer $p > 1$ is prime if $q \nmid p$ for any integer $1 < q \leq \lfloor\sqrt{p}\rfloor$, largest integer less than or equal to $\sqrt{p}$. Thus from the definition, it is clear

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 8, pp: 337-343www.ijaem.net ISSN: 2395-5252

that2 is the only even prime integer, and all other primes are odd integers.
If the difference of pair of primes is two, then that primes are called **twin primes**. For example, 3, 5; 5, 7, etc… are twin primes.Twin primes are supposed to be infinite in numbers [20, 21].

Methods to generate primes:

There are several methods to generate primes and are very simple and efficient. Here we are presenting algorithms of some efficient methods.

1. Sieve of Eratosthenes (Algorithm):

It is the oldest method to extract primes from the finite set of first N natural numbers.
Step-1. Write down all natural numbers up to Nin a table of rows and columns.
Step-2. Delete the first natural number 1 from the table.
Step-3. Search the next non-zero number in the table and delete all those numbers from the table, which are multiples of that number excluding it.
Step-4. Continue Step-4 until the end of the table.
Step-5. Remaining all non-zero numbers in the table are primes.
Step-6. Stop.

2. Sieve of Eratosthenes: (Modified Algorithm)

If $m^2 > N$, then m cannot be the factor of any natural number $\leq N$, and accordingly, theabove algorithm was modified [10-**NET**] that increases the efficiency of the algorithm. It is a fantastic algorithm to find all the primes less than or equal to the given natural numberN.
Step-1. Take a logical array P(N) of N elements and assign all elements true.
Step-2. Assign $t = 2$ and$P(1) = $ false.
Step-3. If $t^2 > N$,then Go to Step-6 otherwise Next Step.
Step-4. If $P(t) =$ true,then make each P(k) false by incrementing k with t from 2t.
Step-5. Add 1 to tand then Go to Step-3
Step-6. The indices of true elements of arrayPare primes.
Step-7. Stop.

A simple Python program for the above algorithm is available on NET [10-NET]. This program gives all primes less than or equal to N with tremendous speed;the only limitation is of computer's memory.

3. Modified Sieve Algorithm (Modified Sieve Method):

Further, we have modified the Sieve of Eratosthenes algorithm to find all primes $\leq (2N + 1)$ for given natural number N. Here we have omitted all even natural numbers, as 2 is the only even prime. Thus, the job reduced to extract primes onlyfrom the odd natural numbers. This one is the most efficient algorithm.
Step-1. Take a logical array P(N) ofN elements and assign all elements true.
Step-2. Assigns $= j = r = 0, t = 1$.
Step-3. If $j \leq N$, then add 1 tos, 2 tot, 4 torandrtoj; otherwise Go to Step-6.
Step-4. If P(s) = true then make each $P(i) =$ false,by incrementing iwith t from j.
Step-5. Go to Step-3.
Step-6. All odd numbers $2i + 1$; $i = 1,2, \ldots, N$ are prime if P(i) is true.
Step-7. Stop.

Knuth (1981), in his monograph, "The Art of Computer Programming" (TAOCP) covers many kinds of programming algorithms with their analysis [12]. Here we are giving a simple computer program in Python to find all primes $\leq (2N + 1)$;

```
def Sieve_of_Thool(n):
# A Python program to find all the primes that are less than or equal 2N+1
    n += 1
    m = n + n
    p = [True for i in range(n)]
    t = c = 1
    k = j = s = r = 0
    while (j < n):
        s += 1
        t += 2
        r += 4
        j += r
        if (p[s]):
            for i in range(j, n, t):
                p[i] = False
    # Printing of all primes less than or equal to 2N+1
```

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 8, pp: 337-343www.ijaem.net ISSN: 2395-5252

```
print(2, end = " ")
for i in range(3, m, 2):
        k += 1
        if (p[k]):
                c += 1
                print(i, end = " ")
print("\nNumber of primes=", c)
```

From the figure Fig-1, it is evident that with the help of modified sieve algorithm, one can extract all primes less than or equal to N, quite rapidly than the sieve of Eratosthenes algorithm. Even then, it has limitations on the input value of N, while applying on computers. We have tried to overcome this difficulty up to a certain extent by developing a new algorithm called **"Farming of Primes".**

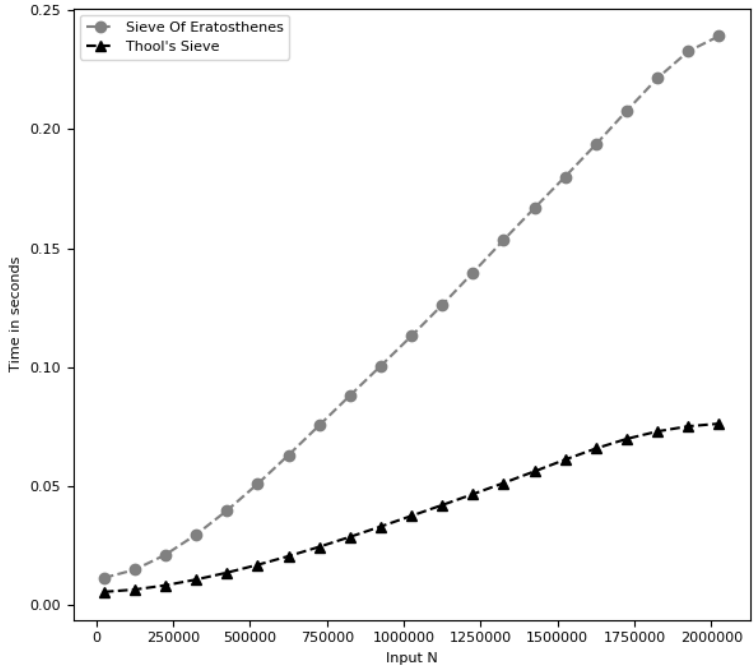

Fig.1 Speed Comparison Graph between Modified Sieve Method and Sieve of Eratosthenes Method

Before giving the actual farming algorithm, firstly, we would like to give our new algorithm to generate primes. It is different from the sieve of Eratosthenes algorithm;

4. A new algorithm to generate the first N prime/s:

Here a novel algorithm is presented to generate required number of primes, which is further used for farming of primes.

Step-1. Take two arrays P and C each of N elements.

Step-2. Assign $v = 3,\ k = 0,\ f = r = t = 1,$ $P(0) = 2.$

Step-3. Perform t-times: Step-4 throughStep-6 provided that$r < N$;otherwise Go to Step-8.

Step-4. For each $j = 1, 2, \dots, k - 1:$ Subtract 1 from $C(j)$ and if $C(j) = 0$ then $f = 0$ and $C(j) = P(j).$

Step-5. If$f = 1$ then assign $P(r) = v$andadd 1 tor otherwise assign $f = 1.$

Step-6. Add 2 to v.

Step-7. Assign $C(k) = 1,$ add 1 tok and assign $t = (P(k)^2 - v)/2$ and thenGo to Step-3.

Step-8. Required prime/s $= P(i): i = 0,1, \dots, r - 1,$ generated by using first k prime/s.

Step-9. Stop.

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 8, pp: 337-343www.ijaem.net ISSN: 2395-5252

Note: First prime $P(0) = 2$. Initially $v = 1 + 2 = 3$ i.e.$P(1) = 3$ is a next prime. To get further primes we are incrementing v by 2 each time and hence we call 2 as a generator of primes.

Proof: After going through the first seven steps, the value of the flag variable f does not change; this indicates that the numbers 3, 5, and 7 (generated by adding two successively in v) are primes. First array element $C(1)$ is get introduced when $v = 9 = (P(1))^2 \Rightarrow P(1)|v$. Second array element $C(2)$ is get introduced when $v = 25 = P(2)^2 \Rightarrow P(2)|v$ and accordingly $(k-1)^{th}$ array element $C(k-1)$ is get introduced when$P(k-1)^2 = v$. When 2 gets added in v, each time 1 gets subtracted from each$C(j)$; $j = 1,2, \dots, k-1$. Thus if $C(j)$ becomes 0 at any juncture, it means that two is added $P(j)$ times in v and hence v is divisible by $P(j)$, and $f = 0$, i.e.$v$is not prime. If none of $C(j)$ become 0, then v is not divisible by any $P(j)$ for $j = 1, 2, \dots, k-1$, and the value of f remains 1. Itmeans that v is prime. That is, only prime numbers are getting stored in an array P. Therefore, to generate first r primes, first $k-1$ prime/s are used.

Note:In this algorithm, without performing actual divisions, we have tested the primality of numbers; hence, it is fast, but it is slower than Eratosthenes methods. With the help of this algorithm, we can verify the actual distribution of primes [8].

Here is a simple module in Python to generate thefirst N primes by using the above algorithm.

```
def Thool_algorithm(n):
# A Python program to generate first N primes
# Define two sufficiently large arrays.
    c = [1 for i in range(1000000)]
    p = [1 for i in range(1000000)]
    p[0] = 2
    v = 3
    r = 1
    k = 0
    flag = True
    while r < n:
        t = p[k] * p[k]
        k += 1
        while v < t:
            for j in range(1, k-1):
                c[j] -= 1
                if c[j] == 0:
                    flag = False
                    c[j] = p[j]
            if flag:
                p[r] = v
                r += 1
                if r == n:
                    break
            else:
                f lag= True
            v += 2
    print ("Required primes are:")
    for i in range(0, n):
        print( p[i] , end = "")
    print("\nThese are generated by using first ", k, " primes only.")
```

5. Farming of Primes (Algorithm):

Let p be the known N^{th} prime.Then one can test the primality of all numbers $< (p+1)^2$or$(p+2)^2$.

Step-1. Preparation: Assign $r = s = k = 0$, $f = t = v = 1$.

Step-2. Sowingof Primes: Assign $P(0) = 2, \dots, P(N-1) = p$ and if $N = 1$ then $P(N) = 3$ else $P(N) = p + 2$.

Step-3. Cultivation of Primes: To plant saplings, take another array Cand assign 1 to each elements.

Step-4. Perform t-times following steps: Step-5 through Step-6.

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 8, pp: 337-343www.ijaem.net ISSN: 2395-5252

Step-5. For each $j = 1$ to$k - 1$: Subtract 1 from each C(j) and if $C(j) = 0$ then assign $C(j) = P(j)$.
Step-6. Add 1 to s, 2 to v and check:If $v > p$ then subtractss from t and Go to Step-8.
Step-7. Add 1 to k and if $k \leq N$, then assign$t = (P(k)^2 - v)/2$, $s = 0$, and then Go to Step-4.
Step-8. Harvesting of Primes: Print P(i) : for $i = 0$ to$N - 1$.
Step-9. Perform t -timesfollowing steps: Step-10throughStep-12.
Step-10. For $j = 1,2, \ldots, k - 1$: Subtract 1 from each C(j) and if $C(j) = 0$ then $f = 0$, $C(j) = P(j)$.
Step-11. If $f = 1$ then print v and add 1 to relse $f = 1$.
Step-12. Add 2 to v.
Step-13. Add 1 to k and if$k \leq N$then $t = (P(k)^2 - v)/2$ and Go to Step-9 otherwise Next Step.
Step-14. The number of additional primes generated = r.
Step-15. Stop.
Note: By modifying above algorithm, one can use it for testing the primality and to find out prime factors of large numbers [17].
Here we are giving a simple module in Python to find all primes generated by first K-primes (less than or equal to 2N) by using the above algorithm;

```
def Farming_of_Primes(n)
# A Python program to find all primes that are generated by primes ≤ 2N.
# Preparation: Initialization of variables.
        flag = True
        t = k = 1
        v = s = r = u = y = z = 0
        b = 3
        m = n + n
        n += 1
        c = [1 for i in range(n)]
        p = [1 for i in range(n)]
        p[0] = 2

# Sowing: By Modified Sieve Method
# Extract all K primes≤ 2Nas seeds to cultivate all primes < (2 + Kthprime)^2.
        while(v<n):
                s += 1
                t += 2
                r += 4
                v += r
                if (p[s] == 1):
                        for i in range (v, n, t):
                                p[i] = 0
        for i in range(3, m, 2):
                u += 1
                if p[u] == 1:
                        p[k] = i
                        k+ = 1
        if k == 1:
                p[k] = 3
        else:
                p[k] = p[k-1]+2

# Cultivation of Primes:
        while(b<m):
                y += 1
                t = p[y]*p[y]
                if t>m:
                        t = m+1
                while b<t:
                        for j in range (1,y):
```

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 8, pp: 337-343www.ijaem.net ISSN: 2395-5252

```
                        c[j] += 1
                        if (c[j] = 0):
                                c[j] = p[j]
                b += 2
# Harvesting of Primes:
        print("\nFollowing are the generated primes:")
        for i in range (0,k):
                print(p[i], end = " ")
        for i in range (y,k+1)
                t = p[i]*p[i]
                while b<t:
                        for j in range (1,i):
                                c[j] += 1
                                if (c[j] == 0):
                                        flag =False
                                        c[j] = p[j]
                        if flag:
                                z += 1
                                print(b, end = " "))
                        else:
                                flag =True
                        b += 2
        print ("\nNumber of additional primes generated =",z)
```

Input value	Number of primes obtained by using		
N	Sieve Method	Modified Sieve Method	Farming Technique
1	0	1	4
2	1	2	9
3	2	3	15
4	2	4	22
5	3	4	22
100	25	46	4236
200	46	78	14615
400	78	139	51967
500	95	168	78359
600	109	196	109066
900	154	278	230738
1000	168	303	283413

Table-1

III. CONCLUSION:

The farming technique surprisingly gives more primes even by using comparatively small-sized arrays. If $P_1, P_2, \ldots, P_N$ $(N > 1)$ are first N primes, then one can generate all the primes less than $(P_N + 2)^2$. Sieve methods discussed earlier extract primes up to a certain extent, whereas in farming techniques, we get extremely more primes if we use all the primes obtained by using sieve methods as our seeds(see Table-1).

While testing the primality of numbers, no divisions are carried out. Hence, the method is fast, efficient and can generate large primes and may be of good support in cryptography, signal processing, and computational Mathematics.

By using the farming technique, one can easily prove that twin primes are infinite in numbers. In addition, it may be helpful to test the primality of Mersenne numbers and to obtain the next Mersenne primes. Therefore, in one sense, it might be helpful to GIMPS members for possibly discovering a record setting, rare, and historic new Mersenne primes. This method is also useful to test the primality and to know the prime factors of a given number.

REFERENCES:

[1]. ANSI X9.31. Public-key cryptography using RSA for the financial services industry.

American National Standard for Financial Services, draft, 1995.

[2]. D. Boneh and M. Franklin. Efficient generation of shared RSA keys. In Advances in Cryptology − CRYPTO'97, vol. 1294 of Lecture Notes in Computer Science, pp. 425–439, Springer-Verlag, 1997.

[3]. J. Brandt and I. Damg˚ard. On generation of probable primes by incremental search. In Advances in Cryptology − CRYPTO'92, vol. 740 of Lecture Notes in Computer Science, pp. 358–370, Springer-Verlag, 1993.

[4]. J. Brandt, I. Damg˚ard, and P. Landrock, Speeding up prime number generation. In Advances in Cryptology − ASIACRYPT'91, vol. 739 of Lecture Notes in Computer Science, 440–449, Springer-Verlag, 1991.

[5]. C. Couvreur and J. Goethals, The RSA public-key cryptosystem, Philips Research Laboratory, Brussels, Report R427, March 1980.

[6]. C. Couvreur and J.-J. Quisquater. An introduction to fast generation of large prime numbers. Philips Journal of Research, vol. 37, pp. 231–264, 1982.

[7]. C. Couvreur and J.-J. Quisquater. An introduction to fast generation of large prime numbers. Philips Journal of Research, vol. 37, pp. 231–264, 1982.

[8]. P. X. Gallagher. On the distribution of primes in short intervals. Mathematica, vol. 23, pp. 4–9, 1976.

[9]. D. Hawkins, 1958. Mathematical sieves. Scientific American 199(December):105-112

[10]. M. Joye and P. Paillier. Fast generation of prime numbers on portable devices: An update. L. Goubin and M. Matsui (Eds.): CHES 2006, LNCS 4249, pp. 160–173, 2006.

[11]. M. Joye, P. Paillier, and S. Vaudenay. Efficient generation of prime numbers. In Cryptographic Hardware and Embedded Systems − CHES 2000, vol. 1965 of Lecture Notes in Computer Science, pp. 340–354, Springer-Verlag, 2000.

[12]. D. E. Knuth. The Art of Computer Programming - Seminumerical Algorithms, vol. 2, Addison-Wesley, 2nd ed., 1981.

[13]. C. Lu and A.L.M. Dos Santos. A note on efficient implementation of prime generation in small portable devices. Computer Networks, vol. 49, pp. 476–491, 2005.

[14]. C. Lu, A.L.M. Dos Santos, and F.R. Pimentel. Implementation of fast RSA key generation on smart cards. In 17th ACM Symposium on Applied Computing, pp. 214–221, ACM Press, 2002.

[15]. U. Maurer. Fast generation of prime numbers and secure public-key cryptographic parameters. Journal of Cryptology, vol. 8, 1995,123–155.

[16]. L. Monier. Evaluation and comparison of two efficient probabilistic primality testing algorithms. Theoretical Computer Science, vol. 12, pp. 97–108, 1980.

[17]. H. Riesel. Prime Numbers and Computer Methods for Factorization, Birkh¨auser,1985.

[18]. R. Rivest, A. Shamir and L. Adieman. A method for obtaining digital signatures and public-key cryptosystems, C. ACM 21,1978, 120-126.

[19]. U. Thool and P. Thul, The Sequence of Primes, Proceedings of International Conference on Mathematical Physics, Nagpur Feb. 2002, India.

[20]. E. W.Weisstein, "Twin Primes", http://mathworld.wolfram.com/TwinPrimes.html

[21]. M. Wolf, The Skewes number for twin primes: counting sign changes of $p_2(x)-C_2 \Li_2(x)$, arXiv:1107.2809.