On the Abundance of Large Primes with Small B-smooth values for p-1: An Aspect of Integer Factorization

Parthajit Roy
Department of Computer Science, The University of Burdwan, West Bengal, India-713104
roy.parthajit@gmail.com

Abstract — The security of RSA public key cryptography and RSA digital signature relies on the assumption of the hardness of the factorization of integers. Since the inception of RSA, there have been a series of proposals for factorizing large integers. A handful of integer factorization methods rely on the fact that the prime factor p having smooth p -1 value. Pollard's p -1 is a method heavily based on this fact. It takes sub-exponential time for certain integers though not suitable for all integers. This paper experiments on the abundance (or scarcity) of smooth p -1 for large primes by examining their availability and suitability.

Keywords — B-Smooth Primes; Number Theory; Integer Factorization; RSA Cryptosystem; Pollard's p -1 Method.

I. INTRODUCTION

Integer factorization is an old technique known to the Greeks before Christ. Though it is known from ancient age, the only technique known was trial and error technique. It is only recently, after the inception of RSA in the year 1978, the subject got an enormous attention.

RSA encryption [1] is a public key encryption system where the product of two large primes is made available as a part of the public key. The model is based on an algebraic structure where the private key can be obtained from the public key if the product of two large primes can be factored.

As there was no practical need of factorization of large numbers before RSA, most of the earlier algorithms were linear to the number of in other words exponential to the bits. All these algorithms are collectively called exponential algorithm in the context of bit representation of the number. i.e. If we need b bits to represent a number, then most of the algorithms would take $O(2^{b/2})$ time for computing the factors which is exponential. Though this is the worst case running time applicable to certain cases, we have to consider only this running time because in case of RSA cryptography always the hardest instance of the problem is selected as to make the system impossible to break by the adversary.

The mid of seventeenth century that the problem got some importance from the mathematics community. Fredric Gauss [2] made a serious effort of integer factorization whose underlying concept has become the bottom line for many modern algorithms. The concept, that Gauss used, was elimination of numbers in trial-division. In his method, a quadratic residue based elimination is used. For a given number there are large numbers of quadratic residues. This eliminates the possibility of certain numbers of being a factor. If a considerable number of quadratic residues are considered, then a huge numbers of integers can be excluded from the trial-division list.

Lehman's proposed a method [3] which is a refinement of Euler's method. In this method, the square difference of two integers has been considered. i.e. n is cleverly expressed as $n = a^2 - b^2 = (a + b)(a - b)$. This states that n is a factor of two integers. The method takes $O(2^{b/3})$ for b bit number. An important fact about Lehman's algorithm is that it takes $O(2^{b/3})$ in most of the cases.

Pollard introduced two important contributions in the fiend of factorization. The methods are called Pollard ρ method [4] and Pollard $p - 1$ method [5]. Pollard $p - 1$ method exploits the smoothness of numbers. The ρ method, however, is a statistical method that generates a sequence of modulo recurrent integers for finding the factors. The ρ method has further been refined by Brent et al [6].

Elliptic curve is another line of attack to the problem of integer factorization proposed by Lenstra [7]. The method inherits the idea of Pollard's $p - 1$ method in the domain of Elliptic curve for factorization.

Number Field Sieve or NFS is another way of factorizing integers. This is a very complicated method that uses Ring based algebraic structure to find the factor integers [8].

Two good review papers on integer factorization are done by Naur [9] [10]. An extensive discussion integer factorization is done by Crandall et al [11] where as a full length book on integer factorization is due to Riesel [12].

Pollard $p-1$ method is one of the most popular and most insightful methods of integer factorization. This paper concentrates on the practical aspects of the Pollard $\rho-1$ method for RSA in particular. The paper shows experimentally that though it is a fascinating method, it cannot be applied for breaking RSA for most of the cases even if no special care has been taken to construct the RSA keys.

The rest of the paper is organized as follows. Section II discusses RSA public key and Pollard $p-1$ method for integer factorization. The section mainly aims the necessary mathematics related to Pollard $p-1$ method i.e. smoothness of numbers. Section III identifies the objective of the proposed research work and the experimental strategy. Section IV presents the results and analyzed them. Conclusion and future scopes come in Section V and references come thereafter.

II. MATHEMATICAL FOUNDATIONS

This section describes the mathematics behind RSA and the integer factorization proposed by Pollard. Though we are talking about Pollard's method, most modern factoring methods considers the smoothness in some way or others. The section also discusses some related mathematical formulae needed to describe the proposed experiment.

A. Mathematics of RSA

RSA cryptosystem [1] is an asymmetric key cryptosystem where two keys, one public and one private key are used. Public key is used for encryption and the private key is used for decryption.

In RSA system, the owner of the key first choses two large primes p and q and computes the product of them as,

$$n = p \times q, \quad \ldots\ldots\ldots \qquad (1)$$

Then the owner computes Euler's totient function of n as follows.

$$\phi(n) = (p-1)(q-1) \quad \ldots\ldots\ldots \quad (2)$$

The owner then chooses some number e so that

$$\gcd\big(e, \phi(n)\big) = 1 \quad \ldots\ldots\ldots \qquad (3)$$

She then computes d such that

$$ed = 1 \bmod \phi(n) \qquad \ldots\ldots \quad (4)$$

And publishes (e, n) as public key and keeps (d, n) secret as private key.

The whole security of the model relies on the question whether the private exponent d can be recovered from the public key or not. Clearly we can recover the private exponent if we somehow know the value of $\phi(n)$. Then using Equation 4, we can recover the value of d. But to get the value of $\phi(n)$ we need to execute Equation 2 and for this, we need $(p-1)$ and $(q-1)$. But the information which is made public is n. So, to get the value of $(p-1)$ and $(q-1)$ one has to factor n which is hard.

B. Pollard's $p-1$ method of factoring

As factoring the product of two large primes is the straight forward way to attack RSA, the whole security of RSA depends on difficulty of factorization. Technically speaking, given n as a b bit integer if all the factoring algorithms take $O(2^{(b/k)})$ time for reasonably small k for all instances, then RSA is secure. i.e. for example, if $k = 10$, then if we consider a prime of 512 bits length, the factoring will take $O\left(2^{\frac{512}{10}}\right) = O(2^{51})$, which is not safe in todays speed of computers. On the other hand, if $k = 5$, then using the same calculations, we get, $O\left(2^{\frac{512}{5}}\right) = O(2^{100})$ which is safe in the present context.

Pollard made a brilliant attempt for factorization in his $p-1$ method. Instead of p and q, in case of the product $n = p.q$, Pollard selected $p-1$. (There is nothing special about $p-1$, we have assumed that p is the smallest among p and q. Otherwise it could be $q-1$ also.) We know from Fermat's theorem that $a^{p-1} = 1 \; mod \; p$ for any +ve $a < p$ where p is prime. Further, any multiple of $p-1$, i.e. if $M = k(p-1)$, then for any a,

$$\begin{aligned} & a^{k(p-1)} \; mod \; p \\ &= (a^{p-1})^k \; mod \; p \\ &= (1^{p-1})^k \; mod \; p \\ &= 1^k \; mod \; p \\ &= 1 \; mod \; p \end{aligned}$$

In other words, if M is multiple of $(p-1)$, then

$$a^M - 1 = 0 \; mod \; p$$

i.e. $p|(a^M - 1)$. Further, if we assume that, $(a^M - 1)$ is not divisible by q, then

$$\gcd(a^{M-1}, n) = p$$

Thus, we can retrieve one of the factors of n. The next part which is more technical is, how shall we compute M. Pollard's brilliant idea was, if $p-1$ is a product of many small primes, then essentially, $p-1$ will divide $u!$ For relatively small value of u [13].

C. Smooth Numbers

So, the whole thing is dependent on whether $p-1$ or $q-1$ are built up of small prime factors or not. This is mathematically called smoothness of an integer. Technically, a positive integer n is said to be $B-Smooth$ if the largest prime factor of n is B. Though there are many *B-Smooth* numbers in the range $(1, n)$, in case of prime numbers this is an important issue. The success of Pollard's $p-1$ method heavily depends on the fact that a prime is a small *B-Smooth* value. This paper typically surveys this fact. This paper tries to investigate whether large primes with small B-Smoothness are abundant or are rare. If there are plentiful of such B-smooth primes, then we need to take a special care for choosing the primes of RSA, otherwise this fact can be ignored.

III. MATERIAL AND METHODS

In the light of the discussions of the previous section, curious minds will definitely ask a question, "*How secure RSA is against Pollard's $p-1$ method?*" This paper addresses this question by experimentation. The paper tries to understand how practical Pollard's $p-1$ method is. i.e. given a product of two integers $n = p \times q$, where p and q are arbitrarily chosen large primes of roughly same length, what is the chance that n can be factored using Pollard's $\rho - 1$ method. The methodology that the paper adapted is as follows.

The paper considers a sequence of prime numbers in valorous range of reasonably large size and tries to factor them using Pollard's $p-1$ method. The results are analyzed using various statistical methods.

The first test that we run is a frequency test. In this test we try to analyze the frequency of primes suitable for factoring using Pollard's $p-1$ method. If there are too many numbers suitable for factoring using this method, then definitely choosing a random prime in a certain range will be vulnerable to such attack. Alternatively, if most of the primes are not suitable for factoring using these method then random selection of primes are safe.

In the present investigation, we have divided the range of smoothness into three parts. First is the range is $B-smooth(p-1) \leq (\log p)^2$. Second range is $(\log n)^2 < B-smooth(p-1) \leq \frac{\sqrt{p}}{2}$ and the third one is $\frac{\sqrt{p}}{2} < B-smooth(p-1)$. In our belief, $(\log p)^2$ is a reasonably handy range to work with. This is because, when we shall consider real primes of RSA, the primes often belong to 1024 bit range. i.e. $\log p = 1024$ and $(\log p)^2 = 1024 \times 1024 = 1\,million$. Means, we will work with factorial of 1000000 and factorial of 1000000 is so large a number that there is a fare chance that both p and q will divide $a^M - 1$ and we will lose our purpose. The second range is even worse than basic range. The ($\frac{\sqrt{p}}{2}$ or more) range is gigantic and the primes are definitely safe primes with respect to factorization using smoothness.

For experiment purpose, we have chosen a very large prime q of 512 bit. q is so chosen that, $q-1$ has a very high B-smoothness. i.e. the largest prime factor of q is far large than the other prime p itself. Typically, the largest prime factor of q is in the range of 100 bits whereas all the testing primes p are less or equal to 60 bits. Thus forcing the prime p to be recovered and not q.

The entire experiment has been run in three different ranges. Namely 40 bit range, 50 bit range and 60 bit range. In each range, we have selected a span of 1 million numbers and 10 such spans. We have run the Pollard's $p-1$ method for factoring the primes in those ranges. Also, computed the B-smoothness of each and every primes in those ranges. We have also counted the B-smooth numbers in different ranges like $B-smooth(p-1) \leq (\log p)^2$ or $(\log n)^2 < B-smooth(p-1) \leq \frac{\sqrt{p}}{2}$ or $\frac{\sqrt{p}}{2} < B-smooth(p-1)$. The entire experiment has been done on Linux platform on Intel core i3 machine. The implementation is done using Python. The outcome of the experiment has been presented in the next section.

IV. RESULT AND ANALYSIS

In this section we will discuss the results and the analysis of the experiments. For the experiment purpose we have the product of two primes $n = p \times q$. We have selected q as a very very large prime in comparison to the other prime p. Also, for the prime q, there are very large prime factor for $q-1$, so that never q will not come out as the factor to be revealed. This setup will force p to come out.

For the purpose of our experiment, we have selected 2^{20} primes in three different ranges. The first range starts from 2^{40}(13 digit primes numbers) and the primes in the range 2^{40} to $2^{40} + 2^{20}$ (i.e. in the range there are one million integers) has been selected. In this way we have selected 10 consecutive ranges (i.e. altogether 10 million integers). The ranges and the numbers are shown in table 1.

TABLE 1 THE COUNT AND PERCENTAGE OF PRIMES IN THE 40 BIT RANGE WHERE THE P-1 HAS A SMALL (LOG-SQUARE) B-SMOOTH VALUES

Start Number	**End Number**	**Total Numbers**	**Total Prime**	$(\log n)^2$ **smooth**	**Percentage (%)**
1099511627776	1099512676352	1048576	37871	454	1.20
1099512676352	1099513724928	1048576	37932	428	1.13
1099513724928	1099514773504	1048576	37823	467	1.23
1099514773504	1099515822080	1048576	37677	451	1.20
1099515822080	1099516870656	1048576	38054	450	1.18
1099516870656	1099517919232	1048576	37864	466	1.23
1099517919232	1099518967808	1048576	37857	423	1.12
1099518967808	1099520016384	1048576	37751	444	1.18
1099520016384	1099521064960	1048576	37782	448	1.19
1099521064960	1099522113536	1048576	37772	424	1.12

It is clear from Table 1, that the primes with small B-smooth values are fairly rare. In our experiment, we have considered, $(\log n)^2$ as the benchmark for the small B-smooth numbers. Only few hundreds such primes are there.The percentage is negligible. Only 1.1% to 1.2% numbers are in this range.

In table 2 the $(\log p)^2$ smooth numbers have been shown in the range of 50 bits numbers. The total number of primes are more or less 30 thousand out of one million integers in every range and the total number of $(\log p)^2$ numbers is less than 100. The percentage of such smooth numbers in this range is between 0.2% to 0.3%.

In table 3 the $(\log p)^2$ smooth numbers have been shown in the range of 60 bits numbers. The total number of primes are more or less 25 thousand out of one million integers in every range and the total number of $(\log p)^2$ numbers is less than 15. The percentage of such smooth numbers in this range is between 0.03% to 0.05%.

Clearly the number of primes with small B-smoothness drastically decreases as the primes become larger and larger. The sharp fall of small B-smooth primes can easily be visualized from figure 1.

TABLE 2 THE COUNT AND PERCENTAGE OF PRIMES IN THE 50 BIT RANGE WHERE THE P-1 HAS A SMALL (LOG-SQUARE) B-SMOOTH VALUES.

Start Number	**End Number**	**Total Numbers**	**Total Prime**	$(\log n)^2$ **smooth**	**Percentage (%)**
1125899906842620	1125899907891200	1048576	30269	90	0.30
1125899907891200	1125899908939770	1048576	30243	67	0.22
1125899908939770	1125899909988350	1048576	30019	59	0.20
1125899909988350	1125899911036920	1048576	30203	55	0.18
1125899911036920	1125899912085500	1048576	30307	66	0.22
1125899912085500	1125899913134080	1048576	30244	85	0.28
1125899913134080	1125899914182650	1048576	29982	63	0.21
1125899914182650	1125899915231230	1048576	30247	66	0.22
1125899915231230	1125899916279800	1048576	30101	68	0.23
1125899916279800	1125899917328380	1048576	30362	72	0.24

TABLE 3 THE COUNT AND PERCENTAGE OF PRIMES IN THE 60 BIT RANGE WHERE THE P-1 HAS A SMALL (LOG-SQUARE) B-SMOOTH VALUES.

Start Number	End Number	Total Numbers	Total Prime	$(\log n)^2$ smooth	Percentage (%)
1152921504606840000	1152921504607890000	1048576	25154	7	0.03
1152921504607890000	1152921504608940000	1048576	24940	13	0.05
1152921504608940000	1152921504609990000	1048576	25103	9	0.04
1152921504609990000	1152921504611040000	1048576	25239	9	0.04
1152921504611040000	1152921504612080000	1048576	25135	13	0.05
1152921504612080000	1152921504613130000	1048576	25459	11	0.04
1152921504613130000	1152921504614180000	1048576	25329	14	0.06
1152921504614180000	1152921504615230000	1048576	25451	11	0.04
1152921504615230000	1152921504616280000	1048576	25276	8	0.03
1152921504616280000	1152921504617330000	1048576	25211	9	0.04

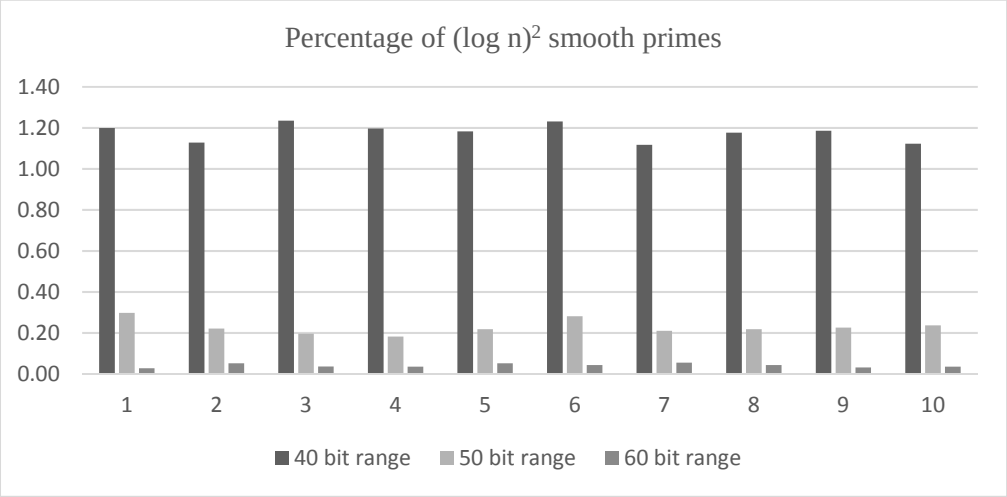

Figure 1 Comparison of log-square smooth p-1 for primes p, in 40 bit 50 bit and 60 bit range. In each range, there are 10 millions of integers. Only primes in the ranges has been considered.

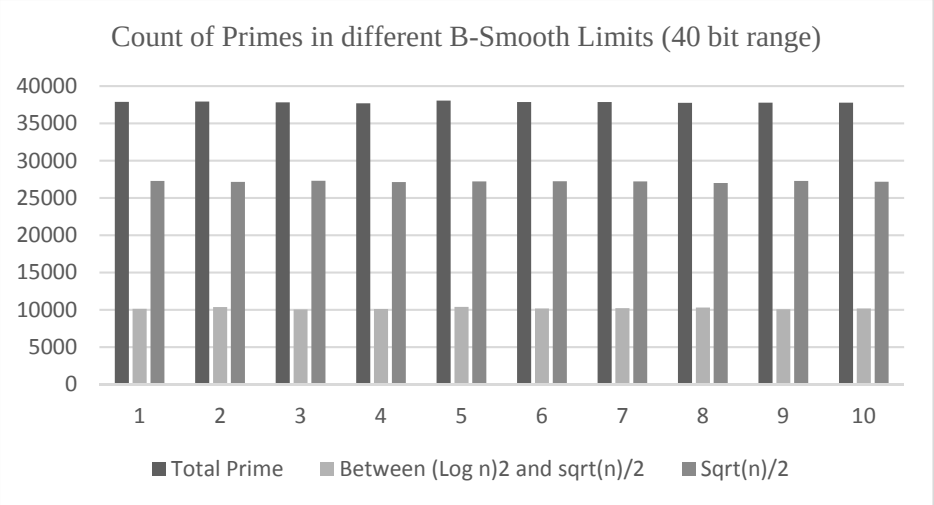

Figure 2 The count of medium and large B-smooth values of p-1 for primes p in 40 bit range. 10 different regions have been chosen in the 40 bit integer range. Each range considers one million integers. The blue bars show the total primes in that range. Orange bars are the count of primes p, where p-1 has moderate B-smooth value and gray bars are the count of primes p, where p-1 has large B-smooth values.

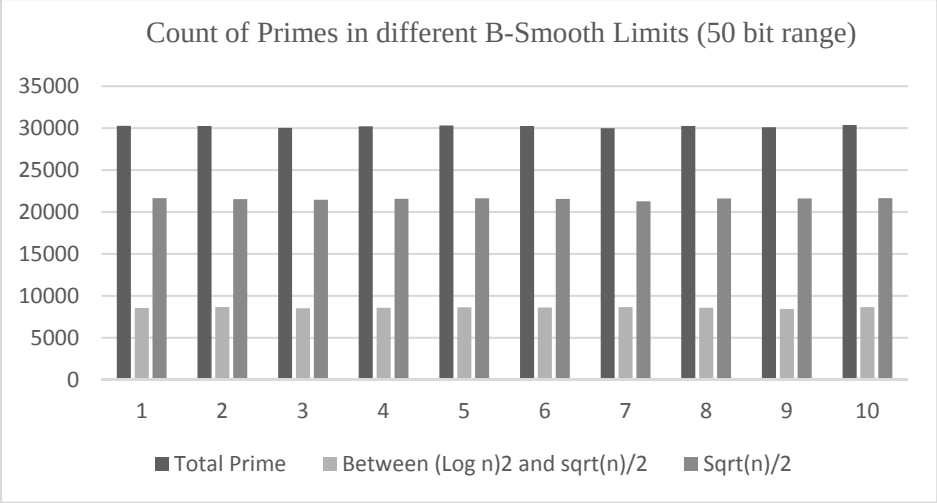

Figure 3 The count of medium and large B-smooth values of p-1 for primes p in 50 bit range. 10 different regions have been chosen in the 50 bit integer range. Each range considers one million integers. The blue bars show the total primes in that range. Orange bars are the count of primes p, where p-1 has moderate B-smooth value and gray bars are the count of primes p, where p-1 has large B-smooth values.

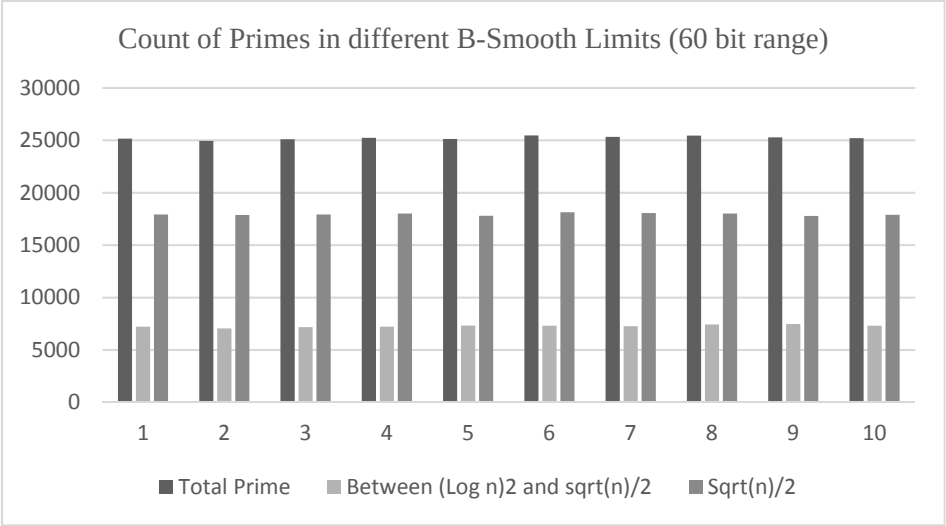

Figure 4 The count of medium and large B-smooth values of p-1 for primes p in 60 bit range. 10 different regions have been chosen in the 60 bit integer range. Each range considers one million integers. The blue bars show the total primes in that range. Orange bars are the count of primes p, where p-1 has moderate B-smooth value and gray bars are the count of primes p, where p-1 has large B-smooth values.

TABLE 4 COMPARATIVE STUDY OF MEDIUM AND LARGE B-SMOOTH VALUES IN THE DOMAIN OF 40 BIT INTEGERS, 50 BIT INTEGERS AND 60 BIT INTEGERS. IN EACH DOMAIN NEAR ABOUT 10 MILLION INTEGERS ARE CONSIDERED. THE PERCENTAGE IS FOR PRIMES ONLY.

Start Number	End Number	Total Numbers	Total Primes	Total Primes with $p-1$ Smoothness $> (\log n)^2$	Percentage (%)
1099511627776	1099522113536	10485760	378383	373928	**98.8226%**
1125899906842620	1125899917328380	10485760	301977	301286	**99.7712%**
1152921504606840000	1152921504617330000	10485760	252297	252193	**99.9588%**

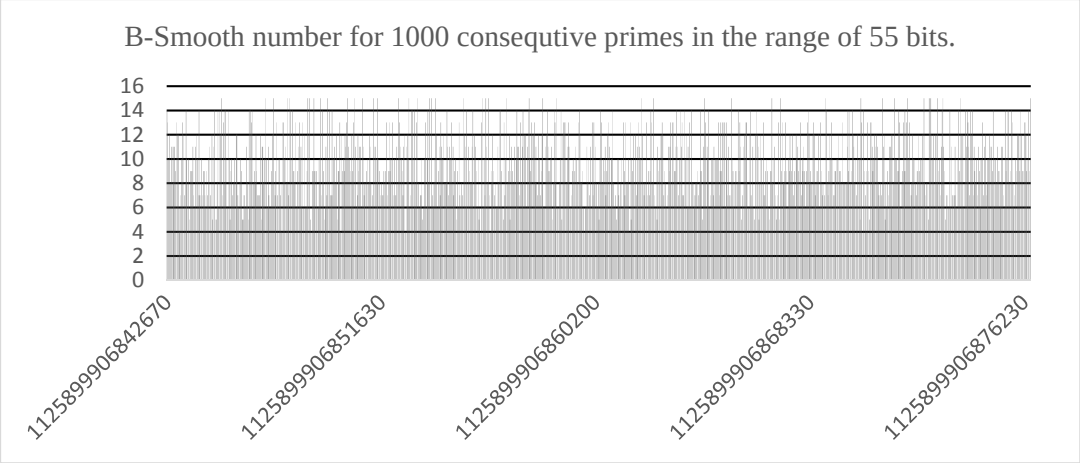

Figure 5 Largest prime factors of p-1 of 1000 consecutive primes in the range of 55 bit numbers.

The distribution of ($\left((log\, n)^2 < B - smooth(p-1) \leq \frac{\sqrt{p}}{2}\right)$ and $(\frac{\sqrt{p}}{2} < B - smooth(p-1))$ has been shown in figure 2, figure 3 and figure 4. In figure 2 the distribution is shown for the primes in the range of 50 bits whereas figure 3 and figure 4 shows the same for 50 bit range primes and 60 bit range primes respectively. It is clear from all the three figures that most of the primes have very large B-smooth value. Also, the together moderate and large B-smooth covers the maximum percentage of primes. Table 4 shows that almost 98.8% 99.7% and 99.9% is the percentage of total integers that have more than $(log\, n)^2$ B-smooth value.

Figure 5 shows the distribution of largest prime factors of 1000 consecutive primes in 55 bit range (18 decimal digit ranges). It shows that most of the B-smooth values have more than 8 digits i.e. roughly 25 bits. Smallest B-smooth values have 4digits i.e. roughly 10 to 12 bits. Also there are many numbers in the range 14 sigits and 15 digits. This means there are many numbers whose largest prime factor is 42 to 45 bits. This is interesting because the numbers are of the range 55 bits.

V. Conclusion and Future scopes

This papers investigates the distribution of the primes in 40 bit, 50 bit and 60 bit ranges having small B-smooth values. The research reveals that the distribution of such primes are very rare. Specially when it goes to higher bit primes it becomes even rarer. The experiment shows that such primes in 60 bit ranges are considerably rare than 50 bit ranges, which is considerably rare than 40 bit ranges. This leads us to conclude when we shall consider 512 bit prime or 1024 bit primes for RSA, such types of primes will be very very rare. They will be so rare that we can simply ignore their occurrences.

Though the present research work got a positive result towards its objective, nevertheless there are scopes of further research. First of all going to even larger range than 60 bit primes will be a good direction. Further, smooth nesses are also used for other types of factorization also. How, smoothness works for elliptic curve based factorization and number field sieve can be experimentally checked to have more concrete conclusion about the rarity of smoothness.

References

[1] R. L. Rivest, A. Shamir and L. Adleman., "A method for obtaining digital signatures and public-key cryptosystems," Communications of the ACM, vol. 21, no. 2, pp. 120-126, February, 1978.
[2] C. Gauss, Disquisitiones Arithmeticae, New Haven: Yale University Press, New Haven, 1966, pp. 329-332.
[3] R. S. Lehman, "Factoring Large Integers," Math. Comp, vol. 28, pp. 637-646, 1974.
[4] J. Pollard, "A Monte Carlo method for factorization," Nordisk Tidskr. Informationsbehandling (BIT), vol. 15, pp. 331-334, 1975.
[5] J. Pollard, "Theorems on factorization and primality testing," Proc. of Cambridge Philos. Soc., vol. 76, pp. 521-528, 1974.
[6] R. Brent, "An Improved Monte Carlo Factorization Algorithm," Nordisk Tidskriftfor Informationsbehandling (BIT), vol. 20, pp. 176-184, 1980.
[7] J. H. W. Lenstra, "Factoring Integers with Elliptic Curves," Ann. of Math, vol. 126, no. 2, pp. 649-673, 1987.
[8] C. Pomerance, "A tale of two sieves," Notices American Mathematical Society, vol. 43, no. 12, pp. 1473-1485, 1996.
[9] T. Naur, lnteger Factorization, DAIMI report, University of Aarhus, 1982.
[10] T. Naur, "New Integer Factorizations," Math. Comp., vol. 41, pp. 687-695, 1983.
[11] R. Crandall and C. Pomerance, Prime Numbers: A Computational Perspective, 2nd ed., Springer Science+Business Media.
[12] H. Riesel, Prime Numbers and Computer Methods for Factorization.
[13] J. Hoffstein, J. Pipher and J. H. Silverman, An Introduction to Mathematical Cryptography, New York: Springer Science+Business Media, 2008.