International Journal of Research in Engineering, Science and Management
Volume 4, Issue 9, September 2021
https://www.ijresm.com | ISSN (Online): 2581-5792

Comparison of Modified Pollard Rho for Discrete Logarithm Problem with the Original

Nagaratna Hegde[1], P. Deepthi[2*]
[1]*Professor, Department of Computer Science and Engineering, Vasavi College of Engineering, Hyderabad, Telangana, India*
[2]*Assistant Professor, Department of Computer Science and Engineering, Bhoj Reddy Engineering College for Women, Hyderabad, Telangana, India*

***Abstract*: Elliptic Curve cryptosystems require small key size to implement public key cryptosystems and appear to be more secure and efficient. The security of Elliptic Curve cryptosystems is based on the difficulty of solving Elliptic Curve Discrete Logarithm Problem (ECDLP). The underlying basis of the many popular Public Key Scheme like Diffie-Hellman and ElGamal is Elliptic Curve Discrete Log Problem (ECDLP). The strength of such public key schemes is predicated on the problem of solving the ECDLP. The best methods for solving ECDLP has time complexity exponential within the size of the underlying field. ECDLP is based on Cryptosystems are popular as they provide good security at key sizes much smaller than number theoretical Public Key Schemes like RSA cryptosystem. ECDLP based cryptosystems are widespread in use, continuous efforts are being done on monitoring the effectiveness of latest attacks or improvements on existing attacks on ECDLP over large field. This paper shows a variant of generic algorithm Pollard's Rho for locating ECDLP using cycle detection with stack and a mix of cycle detection and random walks. Pollard's Rho algorithm using cycle detection with stack requires lesser number of iterations than Pollard's Rho original algorithm in finding the collision. The iteration function used in Pollard's Rho algorithm is not random enough (Knuth, 1969), So Teske proposed a better iteration function by applying more arbitrary multipliers. Random walks allow the iteration function to act randomly than the primary iteration function, thus, the Pollard rho method performs more efficiently than the original. The experiment results show that the proposed methods decrease the number of iterations and speed up the computation of discrete logarithm problem on elliptic curves.**

***Keywords*: Elliptic curves, ECDLP, Pollard's Rho method, Random walks.**

1. Introduction

Diffie-Hellman proposed elliptic curves over finite fields to implement key passing scheme and elliptic curves variants for digital signature. The security of this algorithm is based on the difficulty of solving elliptic curve discrete logarithm problem and if this problem is resolved the cryptosystem is broken. There are several attacks against elliptic curve discrete logarithm problem such as Baby-Step Giant-Step. One of the best algorithms for finding discrete logarithms is Pollard's Rho method to resolve the discrete logarithm problem on general groups, specifically elliptic curve. Automorphism of the iteration function or cycle detection are used to improve Pollard's Rho attack. This paper gives a variant pollard rho algorithm which uses a new cycle detection which was proposed by Nivasch and the random walks proposed by Teske. After that, it analyses the running time of the algorithm and implement the new algorithm. This paper gives some basic definitions for the elliptic curves, Floyd's algorithm and Pollard's Rho algorithm. It describes how Pollard's Rho algorithm may be modified using Nivash's cycle detection instead of Floyd's algorithm. It explains how to introduce random walks on the modified Pollard's Rho and shows how the algorithms are compared.

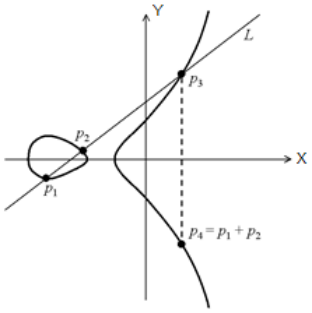

Fig. 1. Point addition (Chord- Tangent rule)

In the field of cryptography elliptic curves plays a very important role. The key size requirement in the traditional public key cryptographic systems such as RSA and ElGamal, are based on difficulty of solving Integer Factorization Problem and Discrete Logarithm Problem (DLP) and are required to be significantly high to achieve desired level of security. Elliptic curve cryptography is based on the difficulty of the Elliptic Curve Discrete Logarithm Problem (ECDLP), where a smaller key size is used in comparison to RSA or ElGamal to achieve same level of security. The smaller key size is used, in the implementation of elliptic curve-based cryptosystems which requires small chip area. The elliptic curve cryptography has become popular in small devices like Smart cards by providing

*Corresponding author: deepthiputnala@gmail.com

good amount of security. Elliptic curve cryptography systems, needs to solve ECDLP in reasonable amount of time. With respect to commutative group an elliptic curve is an abelian group that is, it has multiplication, defined algebraically. The operations of the elliptic curve cryptography are based on the points present on the curve.

The point operation of the chord and tangent law is shown in the figure 1.

Let $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ are two points on the curve, then addition of P and Q can be given by - $P + Q = (x_3, y_3)$

where,

$x_3 = \lambda^2 - x_1 - x_2$

$y_3 = \lambda (x_1 - x_3) - y_1$ and

$\text{Slope}(\lambda) = (y_2 - y_1) / (x_2 - x_1)$ if $P \neq Q$

$\text{Slope}(\lambda) = (3x_1^2 + a) / 2y_1$ if $P = Q$

where, λ is slope of the line passing through the points P and Q and slope of tangent passing through P or Q.

A finite field is a family of elements that necessarily be finite in number. The elliptic curves depend on the finite field. The implementation of finite field arithmetic is an important prerequisite in implementing elliptic curve systems as it is efficient because curve operations are performed using arithmetic operations in the underlying field. Elliptic curve cryptosystem directly relates to the efficiency of solving the ECDLP problem. The elliptic curve cryptosystem is proportional to the order of the finite field which is the strength of the algorithm. The hardness of the cryptosystem increases as the field of the order increases. It is not capable to solve the ECDLP in reasonable amount of time for traditional methods. Pollard's rho algorithm, proposed by J. M. Pollard, gives a good trade-off between time and space to solve the ECDLP problem.

The paper covers the literature review of the work done in solving the ECDLP problem, the background of elliptic curve cryptography which includes brief overview of finite fields, ECDLP problem, and elliptic curves based on the finite fields. Then it gives information about the work done in solving the ECDLP problem. The paper shows the implementation of the Pollard's Rho algorithm.

2. Literature Review

A lot of attacks have been attempted to check the strength of cryptographic systems. In solving the integer factorization problem (IFP), discrete logarithm problem in multiplicative group of finite fields (DLP) and in the group points on an elliptic curve (ECDLP) large amount of work has been carried out. The concept of elliptic curve cryptography was given by V. Miller in 1986. After that N. Koblitz, A. Menezes and S. Vanstone elaborated the concept in 2000. In the papers they proposed a system which uses the elliptic group E(Fq), defined over a finite field Fq as an arithmetic base of cryptosystem. Pollard's Rho method is the strongest known attacks against the ECDLP. J. M. Pollard first proposed the Pollard's Rho method, in 1978. Few improvements were suggested and given by researchers in various papers.

A. Background

1) Arithmetic

Modular arithmetic is the fundamental building block of most public-key cryptologic algorithms. Modular arithmetic in practice hinges on integer arithmetic.

2) Modular arithmetic

Modular arithmetic can be informally thought of as integer arithmetic where the result of any operation is reduced modulo M using least non-negative residues modulo M. Modular arithmetic limits the number of variables in a set, so it is the fundamental of elliptic curve cryptography. To provide a smaller set of numbers modular arithmetic is applied to elliptic curve cryptography with certain properties. All integers from 0 to n-1 are included in the group and cycles around itself and represents only the values of the elements in that group. When divided by n the remainder of numbers represent their values.

For example, the number 9 in Mod 5 represents the value of 4, because five goes into nine once with a remainder of four. 14 is also equivalent to 4, since five goes into 14 twice with a remainder of four. It says that 14 = 4 mod 10.

3) Finite Field

The abstraction of number systems are fields (such as the rational number Q, the real numbers R, and the complex numbers C) and their essential properties. They satisfy the group properties and are abelian in nature which consist of set F together with two operations, addition (+) and multiplication (.). A finite field obey certain rules and is a family of finite number of elements. The prime field, binary field and optimal extension field are three kinds of finite field that are important for efficient implementation of elliptic curve cryptography. The total number of elements present in the field gives the order of a finite field. Efficient implementation of finite field arithmetic is an important factor affecting in elliptic curve cryptosystems.

4) Elliptic curves over finite field

Over a finite field (F_q), Elliptic curve (E) is defined which satisfies the following equation,

$$E: y^2+a_1xy+a_3y=x^3+a_2x^2+a_4x+a_63 \quad (1)$$

where, the equation coefficients are taken from the underlying finite field. The underlying finite field has characteristic not equal to 2 or 3 for the general curve equation which is given by,

$$y^2=x^3+ax+b \quad (2)$$

where $a, b \in F_q$ and $4a^3+27b^2 \neq 0$, together with the special point O at the infinity serving as the identity point for the group. An elliptic curve E is defined over F_q. Let $E(F_q)$be the group of points over the elliptic curve. The number of points on the curve $E(F_q)$, is denoted as $\#E(F_q)$, is called the order of E over F_q. Hasse's theorem is used to give the order of group. Let the two points on the curve be P and Q.

The following group laws are defined for all $P, Q \in E(K)$.

- $P + O = O + P = P$
- $-O = O$
- If $P= (x_1, y_1) \neq O$ then $-P = (-x1, -y1)$

- If Q = -P then P + Q = O
- If P and Q belong to E then P + Q also does.

5) Elliptic Curve Discrete Logarithm Problem (ECDLP)

The hardness of the discrete logarithm problem (DLP) is one of the most important assumptions in modern cryptography. The security of many popular cryptosystems is based on DLP. For example, the Diffie-Hellman key agreement protocol, the ElGamal signature and encryption schemes, the US Government Digital Signature Algorithm (DSA), and the Schnorr signature scheme are such cryptosystems. They worked originally with multiplicative groups of finite prime fields. Koblitz and Miller, proposed elliptic curve cryptosystems by analogous practical systems based on the DLP in groups of points of elliptic curves over finite fields were designed.

Definition: Discrete Logarithm Problem (DLP)

A cyclic group G of prime order p is taken and let $g \in G$ be generator of G. Determine the integer $0 \leq k < p$, given $g, h \in G$, such that $h = g^k$.

Definition: Elliptic Curve Discrete Logarithm Problem (ECDLP)

The discrete logarithm problem in a group G is as follows- given $y \in G$, find an integer x such that, $g^x = y$. An integer x exists if and only if $y \in \langle g \rangle$, where $\langle g \rangle$ is the cyclic subgroup of g generated by g. Let an elliptic curve $E(F_q)$ be defined over a finite field F_q of order p. Let the two points on the elliptic curve P and Q, such that $P, Q \in E(F_q)$. Point P is the generator point and forms a cyclic group <P>. Point Q is the point in the group formed by P such that $Q \in \langle P \rangle$. The elliptic curve discrete logarithm problem (ECDLP) problem is to find an integer l such a that,

$$Q = lP = (P + P + \ldots. + l \text{ times}$$

In the elliptic curve of finite field F_q, the operations are defined over addition operation and the point doubling.

For solving the ECDLP problem many attacks have been proposed. The naïve approach is the most basic approach which is also known as exhaustive search. The sequence of points P, 2P, 3P, 4P, …., one computes until it encounters Q. The running time is n steps in the worst case and n/2 steps in average case approximately. By selecting elliptic curve parameters sufficiently large exhaustive search can be circumvented.

3. Floyd's Cycle-finding Algorithm

It is better to choose Floyd's algorithm to minimize the memory requirement and running time, instead of comparing each new Y_i to all previous ones and store all elements until obtaining collision. One computes pairs (Y_i, Y_{2i}) of points for i =1,2,3... until finding $Y_i = Y_{2i}$. The previous pair can be discarded, after computing a new pair, thus the storage requirements are negligible.

4. Pollard's Rho Algorithm

In this algorithm three possibilities are chosen in a random manner and the resulting sequence is sufficiently complicated to be regarded as a random mapping is the idea of Pollard. First start with R_0, the random point and build the sequence R_i with the iteration function f until the collision occurs. In fact, the sequence R_i become periodic after some iterations, $E(F_p)$ is finite, so there will be some indices $i<j$ such that $R_i = R_j$, j-i is the period and R_i, $R_i+1, R_i+2, \ldots, R_j$ form a loop. For cycle detection, Floyd's method is used. The original Pollard's Rho method on elliptic curves is given below:

- Split $E(F_p)$ into three disjoint sets S_1, S_2 and S_3 of roughly equal size
- Let $R_0 = a_0P + b_0Q$ with a_0 and b_0 two random integers in]0, n[and the iterative function f was defined as:

Algorithm 1. Iteration function

1: function f (R): R_1
2: if R Î S_1 then
3: $R_1 \leftarrow R + P$
4: else if R Î S_2 then
5: $R_1 \leftarrow 2R$
6: else
7: $R_1 \leftarrow R + Q$
8: end if
9: return R_1
10: end function
11: function f (a, b): a_1, b_1
12: if R Î S_1 then
13: $a_1 \leftarrow a + 1$
14: else if R Î S_2 then
15: $a_1 \leftarrow 2a$
16: $b_1 \leftarrow 2b$
17: else
18: $b_1 \leftarrow b + 1$
19: end if
20: return a_1, b_1
21: end function

Algorithm 2. Pollard's Rho with Floyd's cycle finding algorithm

Require: P, Q, S_1, S_2, S_3
Ensure: Integer l where Q = lP
1: $a_0 \leftarrow$ random Î]0; n[
2: $b_0 \leftarrow$ random Î]0; n[
3: $j \leftarrow 0$
4: $R_0 \leftarrow a_0P+b_0Q$
5: for all j such that $R_j \neq R_{2j}$ do
6: $(R_{j+1}, a_{j+1}, b_{j+1}) \neg f(R_j), f(a_j, b_j)$
7: $(R_{2(j+1)}, a_{2(j+1)}, b_{2(j+1)}) \leftarrow f(f(R_{2j})), f(f(a_{2j}, b_{2j}))$
8: $j \leftarrow j + 1$
9: if $R_j = R_{2j}$ and $b_j \neq b_{2j}$ then

$$l \leftarrow \frac{a_{2j} - a_j}{b_j - b_{2j}} \bmod(n)$$

10: else if $b_j = b_{2j}$ then
11: $a_0 \leftarrow$ random Î 0; n[
12: $b_0 \leftarrow$ random Î]0; n[
13: $j \leftarrow 0$

14: end if
15: end for
16: Return l

The Pollard's Rho algorithm is known as the best algorithm to resolve ECDLP in the generic groups. The memory requirement is negligible, and the running time is exponential.

5. Pollard's Rho Algorithm with Random Walks

The iteration function used in Pollard's Rho algorithm is not random enough (Knuth, 1969), So Teske proposed a better iteration function by applying more arbitrary multipliers. Divide $E(F_p)$ into s disjoint subsets $S_1, S_2,..,S_s$ of approximately the same size. The good choice for s should be around 20 (Teske, 2001). Choose random integers a_i, b_i mod n.

Let $S_i = \{R(X,Y) \hat{I} E(F_p)\}$ $X(\text{mod } s) = i\}$ and $M_i = a_iP + b_iQ$ So the iteration function is defined as below:

$$f(R_j)=R_j+1=M_i+R_j \text{ if } R_j \hat{I} S_i$$

The pseudocode of the iteration function is:

Algorithm 3. Iteration function
1: function f (R): R_1
2: if R Î S_i then
3: $R_1 \leftarrow R + M_i$
4: end if
5: return R_1
6: end function
7: function f (R, a, b): a_1, b_1
8: if R Î S_i then
9: $a_1 \leftarrow a_i + a$
10: $b_1 \leftarrow b_i + b$
11: end if
12: return a_1, b_1
13: end function

The Pollard Rho original can be modified just by replacing the old iteration function by the new one.

6. Pollard's Rho Algorithm Using Stack

This section explains how Pollard's Rho method using stack is improved. First, Nivasch's method for cycle detection (Nivasch, 2004) is described. Second, the Pollard's Rho modified is outlined and followed by methods with random walks. Then all are compared and the best one is selected. Finally, the proposed algorithm is implemented and make a comparison with the original algorithm.

Nivasch's Cycle-finding Algorithm:

First stack should be created and starts out to be empty. At each step j, all the top entries (x_i, i) (pop) are removed from the stack where $x_i > x_j$. If $x_i = x_j$ is found in the stack, it is completed. The length of the cycle is j-i. Else, to the top of the stack (push), add (x_j, j) and continue. This algorithm uses logarithmic space and halt in the smallest value of the sequence cycle and run-in linear time.

7. Pollard's Rho Algorithm Modified

The idea is to use the iteration function for generating and storing the elements in the stack and to use stack, the main algorithm is as follows: keep in the stack pairs (i, a_i, b_i, R_i), the R_i it defines lexicography order on $E(F_p)$, in the stack forms increasing sequence. If $R_j > R_i$ where $0<i<j$, then push the pairs (j, a_j, b_j, R_j) in the stack, otherwise if $R_i > R_j$ then pop all pairs (i, a_i, b_i, R_i) until we find $R_i = R_j$ or $R_i < R_j$. In the first case, halt the process and compute l. In the second case, push (j, a_j, b_j, R_j) in the stack and generate other points.

The detailed procedure is as follows:

- Three disjoint sets S_1, S_2 and S_3 are taken from $E(F_p)$
- S_1, S_2 and S_3 contain points with y-coordinate value between [0, p/3[, [p/3, 2p/3[or [2p/3, p [successively
- Let $R_0 = a_0P + b_0Q$ with a_0 and b_0 two random integers Î [1, n-1] and we define the iterative function f

The a_i and b_i sequence can be computed as follow:

If R_j is less than R_i pop from the stack all R_i where $R_j < R_i$ else push R_j on the top of the stack and continue. The process is halted when $R_i = R_j$ then find the fixed point, the logarithm discrete is resolved and

The pseudo-code is as follow:
Require: P, Q, S_1, S_2, S_3
Ensure: Integer l such that Q = lP
1: $a_0 \leftarrow$ random Î]0; n[
2: $b_0 \leftarrow$ random Î]0; n[
3: j ← 1
4: i ← 0
5: $R_0 \leftarrow a_0P + b_0Q$
6: $(R_j, a_j, b_j) \neg (f(R_0), f(a_0, b_0))$
7: stack ← push (i, a_i, b_i, R_i)
8: for all j such that $R_j \neq R_i$ do
9: if $R_j < R_i$ then
10: repeat
11: stack.pop()
12: stack:top() ¬ $(i; a_i; b_i; R_i)$
13: until $R_j \geq R_i$
14: end if
15: if $R_j > R_i$ then
16: stack ← push(j, a_j, b_j, R_j)
17: $(R_{j+1}, a_{j+1}, b_{j+1}) \leftarrow (f(R_j), f(a_j, b_j))$
18: else

$$l \leftarrow \frac{a_j - a_i}{b_i - b_j} \text{mod}(n)$$

19: break
20: end if
21: j ← j + 1
22: end for
23: Return l

The stack size must not exceed the memory allocated for this attack. In that case, this attack will be discarded and unused.

Random Walks:

Random walks which were developed by Teske are known to do much better than the original iteration function at the cost of little more computation. Hence, use a mixture of random walks and stack in the Pollard's Rho algorithm. The steps in

Pollard's Rho modified algorithm is described as:

- Disjoint sets $S_1, S_2,.. S_{30}$ of $E(F_p)$ is partitioned into 30 sets
- $S_1, S_2,\ldots,S_{30}$ contain points with x-coordinate mod 30 equals to 0, 1, 2, …29 successively
- Generate 30 random couples (a_i, b_i) and compute the points $M_i = a_i P + b_i Q$
- Let $R_0 = a_0P+b_0Q$ with a_0 and b_0 two random integers in [1, n-1] and define the iterative function f:

$$f(R_i)=R_{i+1}=\{M_j+R_i \text{ if } R_i \hat{I} S_j\}$$

The a_i and b_i sequence can be computed as follow:

$$(a_i,b_i)=\{(a+a_j,b+b_j)\}\text{if } R_i \hat{I} S_j$$

- If R_{i+1} is less than R_i pop from the stack all R_i where R_{i+1} is the least else push R_{i+1} on the top of the stack and continue
- Halt the process when $R_i = R_k$ then find the fixed point, the logarithm discrete is resolved and

$$l = \frac{a_k - a_i}{b_i - b_k} (\text{mod } n)$$

8. Research

The research was started by finding information about elliptic curves since they are the base of this paper. The articles that laid the foundation for the knowledge of Pollard's rho method was read from the original articles given by Pollard. The information about Pollard's rho method for discrete logarithm problem was known. Since this paper is about Pollard's rho method for elliptic curve discrete logarithm problem, all the articles and books were of importance which are related to them. Later, when the first tests were done, some articles written by Teske were studied to understand the steps for testing.

A. Implementation

The algorithms were executed in C, using GNU multi precision (GNU MP) libraries. GNU MP is a portable library which is used to carry out the computations on large numbers (like arbitrary precision integer and rational number arithmetic). It is basically used to provide the fastest possible arithmetic for all applications like cryptography. For all applications GNU MP give good performance and is designed for choosing algorithms based on the sizes of the operands. As most applications tend to use just a few words of precision; but some applications (Like Cryptography) need thousands of words, for which GNU MP can be used. The code is original except for some implementations of elliptic curve functions. The Pollard's rho method is implemented in C with different iterating functions.

The Pollard's Rho has been tested using stack as:

By first, producing file containing p prime numbers and for each prime number a secure elliptic curve id builds and choose arbitrary point P and Q, then add the X and Y coordinate of P and Q to the file. Second, use this approach to compute the discrete logarithm of Q to the base P. Follow these steps to implement the algorithm:

- Generate p prime numbers with size between fifteen and twenty digits (ten generations for each size)
- Generate random numbers A and B such that $(4a^3 + 27b^2)$ mod (p) different from 0
- Use A and B numbers, p prime numbers, to generate elliptic curve
- For Points P and Q choose random X coordinate from $E(F_p)$ and calculate Y coordinate using Weierstrass Equation
- Compute integer n such that Q = nP using the method described above

9. Comparison between Algorithms

Analyzing the algorithms is quite important in computer programming because there are usually several algorithms available for a particular application and one would like to know which is the best.

A. Analysis

It's quite important to compute the number of evaluations to analyze the performance of the modified method which is the most expensive steps in the Pollard's Rho method is the evaluation of the iteration function. The number of iterations in Pollard's Rho modified is less when compared to original Pollard's Rho. At each iteration, f has been evaluated three times instead of one time with the Pollard's Rho modified. It concludes that the running time will be the greatest in the Pollard's Rho original. The memory for storage is more for the Pollard's Rho original because it stores only pair (Y_i, Y_{2i}) and in each step, it generates the new pairs and the memory used is negligeable as it discards the previous pairs. Pollard's Rho modified requires more memory as it stores all generated R_i to form an increasing sequence and then delete the element from the stack only if the current R_i is less than this element.

10. Experiment Results

Several experiments are done on random elliptic curves by computing the running time and the number of evaluations of iteration function for each of them. In Table 1, it was found that if the length of p is less than twenty digits, the running time of Pollard's Rho modified is less than the original Pollard's Rho and the number of iterations is compared with the iteration function of two methods, it is the lowest in Pollard's Rho with stack. By this the Pollard Rho's modified method performs better than the original Pollard's Rho method. In Table 2, the running time of Pollard's Rho is compared with the random walks and Pollard's Rho mixing stack and random walks. In the experiment prime numbers are used with size between fifteen and twenty digits and the experimental results prove that the first method is lower than the second. It was found that mixed walks using stack perform better than Pollard's Rho only with random walks.

Table 1
Running time comparison and Number of iterations of new Pollard's RHO and Original Pollard's RHO

No. of Digits	Pollard Rho with Stack		Pollard Rho with Floyds Cycle	
	Time (Seconds)	Number of Iterations	Time (Seconds)	Number of Iterations
15	0.143455	1020	0.078954	489
16	0.456321	3456	0.231233	2134
17	1.234563	123124	1.013451	54367
18	4.543781	378579	3.903451	23410
19	6.333490	543671	4.653567	412389
20	8.908321	1321456	7.454322	567834

Table 2
Running time of Pollard's Rho with random walks and mixing stack and random walks

No. of Digits	Pollard Rho with Stack	Pollard Rho with Floyds Cycle
15	2.134546	1.023123
16	15.231456	10.342356
17	28.345234	25.760982
18	123.435246	101.234896
19	345.236123	231.345890
20	2314.543982	1212.342123

11. Conclusion

This study outlined a new algorithm to speed Pollard's Rho attack by making use of first Nivasch's cycle detection and second mixing Nivasch's cycle detection and random walks. It is not appropriate if the memory used is limited. For further research, it can be intended to improve Pollard's Rho method using the same time cycle detection, random walks and Parallelization.

12. Future Scope

Further research, can be done by Parallelizing Pollard's Rho method using at the same time cycle detection, random walks.

References

[1] G. Nivasch, "Cycle detection using a stack", Information Processing Letters 90/3, pp. 135-140.
[2] Aritro Sengupta and Utpal Kumar Ray, "Message mapping and reverse mapping in elliptic curve cryptosystem", Security and communication networks, 2016; 9:5363–5375.
[3] Hankerson, Darrel, Menezes, Alfred J., Vanstone, Scott, "Guide to Elliptic Curve Cryptography."
[4] Edlyn Teske. Speeding up pollard's rho method for computing discrete logarithms. In International Algorithmic Number Theory Symposium, pp. 541– 554. Springer, 1998.
[5] Edlyn Teske. On random walks for pollard's rho method. Mathematics of computation, 70(234):809–825, 2000.
[6] John M Pollard. A monte carlo method for factorization. BIT Numerical Mathematics, 15(3):331–334, 1975.
[7] John M Pollard. Monte carlo methods for index computation (modp), Mathematics of computation, 32(143):918–924, 1978.
[8] Victor S Miller. Use of elliptic curves in cryptography. In Conference on the theory and application of cryptographic techniques, pages 417–426. Springer, 1985.
[9] Jenny Falk. "On Pollard's rho method for solving the elliptic curve discrete logarithm problem", Department of Mathematics, Linnaeus University, 2019
[10] Siham Ezzouak, Mohammed Elamrani and Abdelmalek Azizi," A variant of pollard's rho attack on elliptic curve cryptosystems" Journal of Computer Science 10 (8): 1575-1581, 2014.