Online at: https://researchtrendsjournal.com
Indexed Journal

ISSN No: 2584-282X
Peer Reviewed Journal

INTERNATIONAL JOURNAL OF TRENDS IN EMERGING RESEARCH AND DEVELOPMENT

Volume 3; Issue 1; 2025; Page No. 224-231

Received: 17-10-2024
Accepted: 26-11-2024

Cryptographic Hardness Assumptions Based on Number-Theoretic Problems

[1]Tripti Gautam and [2]Dr. Narendra Bahadur Singh

[1]Research Scholar, Mahakaushal University, Jabalpur, Madhya Pradesh, India
[2]Professor, Mahakaushal University, Jabalpur, Madhya Pradesh, India

DOI: https://doi.org/10.5281/zenodo.18384065

Corresponding Author: Tripti Gautam

Abstract

This paper explores the role of number theoretic methods in cryptography, emphasizing the use of primitive Pythagorean triples for developing new encryption and decryption algorithms through a fundamental Pythagorean tree. It reviews computational number theory, focusing on primality testing and various tests like Miller-Rabin and elliptic curve, which enhance algorithm efficiency. The discussion includes references to works by Silverman and Tate, and Hankerson *et al*., highlighting implementation challenges in elliptic-curve cryptography. The paper asserts the importance of number theoretic approaches and critiques existing algorithms for their speed and security limitations, noting ongoing research efforts toward improved solutions.

Keywords: Applications, Number Theory, Cryptography, digital, communication

Introduction

Mathematical methods for protecting data and communications are the focus of cryptography. By doing this, we can make sure that only authorized parties may decipher the original data by converting it from plaintext to ciphertext and back again. Protecting private information while ensuring its authenticity and non-repudiation are the key goals of cryptography. Protecting private information and conversations in today's hyper-digital environment from cybercriminals requires these goals. Cryptography is a crucial component of contemporary computer systems, enabling safe communication in domains including online banking, government, e-commerce, and personal data security. Countless sensitive information, including login credentials, financial data, digital money, emails, and more, rely on cryptographic techniques for protection. There are two primary subfields of cryptography: symmetric-key cryptography, in which one key is used for both encryption and decryption, and asymmetric-key cryptography, in which two keys, one public and one private, are utilized. By eliminating the need for a shared secret key in secure communication, public-key cryptography (PKC), which emerged in the 1970s, radically altered the landscape of security. The difficulty of certain mathematical problems-like calculating elliptic curve discrete logarithms, factoring big numbers, and solving discrete logarithms-is the basis for the strength of cryptographic systems. Staying ahead of possible security risks requires cryptography approaches to change with technology and computational capacity. The building blocks of digital security, cryptography provide confidentiality, safe financial dealings, and confidence in online platforms.

When it comes to safeguarding sensitive information from prying eyes or manipulation, cryptography is the way to go. It has grown in importance to become an essential part of many things, including private internet interactions, digital transactions, and confidential conversations. Data transmission across the internet and other potentially unsecure networks requires cryptography to guarantee the data's validity, integrity, and secrecy. Secure encryption methods and protocols have their mathematical roots in number theory, a subfield of mathematics that is fundamental to cryptography. Examining integers and their connections, with special emphasis on prime numbers, modular arithmetic, GCDs, and Diophantine equations, is the main topic. These ideas are used in cryptography to

build systems that can protect information well. Cryptographic techniques guarantee that confidential information may only be accessed by authorized individuals by capitalizing on the complexity of certain number-theoretical problems, such as computing discrete logarithms or factoring substantial integers. Digitized signatures and public-key cryptography, which employs asymmetric encryption, are examples of strong security protocols made possible by the interaction between cryptography and number theory. With the mathematical guarantee of security provided by the intrinsic difficulty of solving number-theoretic issues, these cryptographic systems are able to function.

Literature Review

Hamlin, Nathan. (2017) [1]. We need a representation and number analysis up to the challenge of quantum computing. In this paper, we clarify key aspects of representation with respect to combinatorics, which are important for understanding digital integer representations and, by implication, for understanding numbers and their practical and theoretical uses in mathematics. Specifically, the author is crossing their fingers that this research may clarify several mysteries surrounding lattice-based codes like the Generalized Knapsack Code- and its potential applications to computing for mathematicians and computer scientists.

Saranya, C. & Janaki, G. (2021) [2]. Among the many fields that can benefit from an understanding of number theory, this chapter will go over a few examples from cryptography, statics, and chemistry. Specifically, it will go over how to use the Raising Sun Cryptography Technique to identify enciphering exponent and recovery element, the linear Diophantine equation for chemical equation balancing utilizing matrices, and the Pythagorean triangle and the Jarasandha numbers for static ladder issues.

Annu, (2024) [3]. Encryption techniques developed in the contemporary era that rely on Data security and veracity are guaranteed via algebraic structures. The focus of this study is on groups, rings, fields, and lattices-the basic algebraic structures upon which modern cryptographic systems are based. Among other cryptographic methods and protocols, we examine these structures in detail and show how they may be used for encryption using public keys, digital signatures, and hash algorithms. an overview of algebra's key ideas and characteristics, then a study of its practical uses in cryptography. By using elliptic curve groups, Elliptic Curve Cryptography (ECC) outperforms traditional approaches like RSA in terms of security while requiring fewer key sizes. In addition, there are promising possibilities for post-quantum security using lattice-based encryption, which takes use of the difficulties in order to thwart assaults by quantum computers using lattice difficulties.

Asole, R. *et al.* (2025) [4]. The mathematical foundation for contemporary cryptography, which includes group theory, is crucial for secure communication, authentication, and encryption. All sorts digital signatures, symmetric and asymmetric encryption, and other cryptographic methods, key exchange systems, and group theory, and other related topics, is investigated in this work. Computational challenges including particular focus is placed on integer factorization, the discrete logarithm issue, elliptic curve encryption, and cyclic groups. We promise that all cryptographic protocols used in the real world are secure and efficient. applications by learning about the algebraic features of groups.

Easttom, William. (2021) [5]. This textbook provides a thorough introduction to cryptography, including on the one hand, and real-world applications, on the other. Applying the informational resources offered in each chapter, the book applies cryptography to real-world security scenarios. Dr. Chuck Easttom, a prolific author, sets forth the groundwork for modern data security by outlining necessary mathematical abilities and providing a comprehensive explanation of how to use cryptographic algorithms. Cryptography 101 teaches and practices a variety of cryptographic techniques, including hashes and ciphers, key generation, virtual private networks (VPNs), and web, email, and voice communications encryption.

Research Methodology

One of the two major issues with computing prime numbers is primality testing, which may have been first investigated by Euclid almost two millennia ago, but it wasn't until 1801 that Gauss acknowledged it as a major problem. We will examine a number of current techniques for primality testing in this chapter, including

- A few elementary primality tests, often executed in exponential time *EXP*.
- Random polynomial time is used by the Rabin-Miller assay *RP*.
- Probabilistic elliptic curve tests with zero margins of error use polynomial-time may be executed *ZPP*.
- Polynomial time is used for the AKS test and is deterministic *P*.

Basic Tests

What follows is an explanation of the Primality Test Problem (PTP):

$$PTP := \begin{cases} input: & n \in \mathbb{Z}_{>1} \\ output: & \begin{cases} Yes: & n \in primes \\ No: & Otherwise \end{cases} \end{cases}$$

An important and widely-known theorem in primality testing is this one:

Theorem 1: Let $n > 1$. Assuming n does not divide evenly into any prime numbers between $\lfloor \sqrt{n} \rfloor$, hence, the integer n must be prime.

The easiest one metric to use to determine whether n is primality is all the prime factors up to $\lfloor \sqrt{n} \rfloor$, Here is the test: we employ the Eratosthenes Screen; it finds all prime integers larger than n. Trial divisions for primality testing:

$$Test\ (pi) \overset{def}{=} p_1, p_2, \ldots, p_k \leq \lfloor \sqrt{n} \rfloor, p_i \nmid n$$
$$\Uparrow$$
$$Eratosthenes\ sieve$$

So, if n is passed by Test p_i, hence, the integer n must be prime:

$$n\ passes\ Test\ (p_i) \Rightarrow n \in primes.$$

Example 1: It is sufficient to examine the primes up to in order to determine whether 3271 is prime $\lfloor\sqrt{3271}\rfloor = 57$. That is, the following will suffice as the maximum number of trial divisions:

$$\frac{3271}{2}, \frac{3271}{3}, \frac{3271}{5}, \frac{3271}{7}, \ldots, \frac{3271}{47}, \frac{3271}{53}.$$

The fact that 3271 does not divide to zero proves that it constitutes a prime number. With that being stated, n=3273, normally we plan to conduct the following trial divisions:

$$\frac{3273}{2}, \frac{3273}{3}, \frac{3273}{5}, \frac{3273}{7}, \ldots, \frac{3273}{47}, \frac{3273}{53}.$$

Luckily, we can skip doing every one of the trial sections, since 3273 is a composite. As the trial is over, really divisions, $\frac{3273}{3}$, There is no way that 3273 is not a composite number since this is a remainder of zero.
Despite its simplicity, this test is impractical when dealing with big numbers since it requires $\mathcal{O}(2^{\log n)/2})$.

Theorem 2 (The inverse in 1876, Lucas asserted that the small theorem of Fermat Let $n > 1$. So, let's pretend that there's an integer a that serves as the initial value of n, which signifies that

1. $a^{n-1} \equiv 1(mod\ n)$
2. $a^{(n-1)/p} \not\equiv 1(mod\ n)$, for eachprime divisor p of $n - 1$.

Next, n must be a prime number.

Proof: Since $a^{n-1} \equiv 1(mod\ n)$, As stated in the first part of the theorem, $\text{ord}_n(a) \mid (n-1)$. We will show that $\text{ord}_n(a) = (n-1)$. Suppose $\text{ord}_n(a) \neq (n-1)$. Since $\text{ord}_n(a) \mid (n-1)$, For every positive integer k, there is an integer that $n - 1 = k.\text{ord}_n(a)$ Since $\text{ord}_n(a) \neq (n-1)$, By assuming p is prime, we may deduce that $k > 1$. factor of k. It follows that

$$\frac{x^{n-1}}{q} = x^{\frac{k}{q}.ord_n(a)} = \left(x^{ord_n(a)}\right)^{\frac{k}{q}} \equiv 1(mod\ n).$$

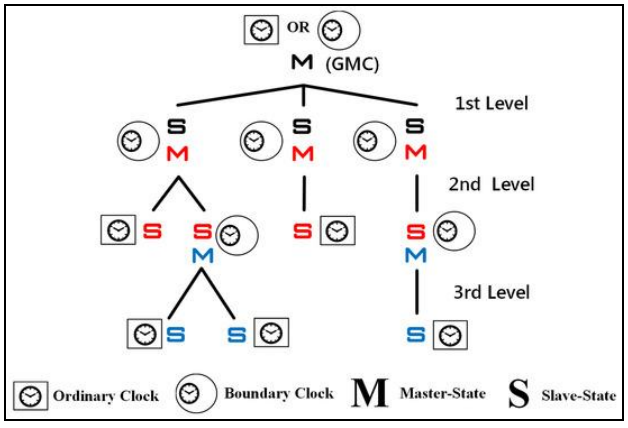

Fig 1: Procedures and Programs for PTP

But this is counter to the theorem's postulate, thus we'll need to $\text{ord}_n(a) = (n-1)$. Okay, so, ever since $\text{ord}_n(a) \leq \varphi(n)$ and $\varphi(n) \leq n - 1$, it follows that $\varphi(n) = n - 1$.
Here is how to rigorously verify primality using Lucas' theorem:

Test for primality using ancestral forms:

$$Test\ (a) \overset{def}{=} a^{n-1} \equiv 1(mod\ n)$$

$$a^{(n-1)/p} \not\equiv 1(mod\ n), \forall p|(n-2)$$

If *n* is found to be prime, then:

$$n\ passes\ Test\ (a) \Rightarrow n\ \in primes.$$

Testing for primality via ancestral forms is also referred to - as n − 1 primality test, using prime factorization as its foundation $n - 1$.

Example 2: Assuming n=2011, we get 2011-1=2·3 ¦5 ·67. Remember that the lowest value of 2011 is represented by the third digit, which is a primitive root, since the order (3,2011) = φ (2011) = 2010. This means that

$$3^{2011-1} \equiv 1\ (mod\ 2011),$$

$$3^{(2011-1)/2} \equiv -1 \not\equiv 1(mod\ 2011),$$

$$3^{(2011-1)/3} \equiv 205 \not\equiv 1(mod\ 2011),$$

$$3^{(2011-1)/5} \equiv 1328 \not\equiv 1(mod\ 2011),$$

$$3^{(2011-1)/67} \equiv 116 \not\equiv 1(mod\ 2011),$$

Number Theory and Cryptographic Hardness Assumptions

Preliminaries and Basic Group Theory

First, we will go over the fundamentals of modular arithmetic and prime numbers. Readers familiar with the subjects should just need to scan the next two parts, since they include evidence for the majority of the claimed outcomes and may contain some new information.

Primes and Divisibility

Regarding numbers, Z For all integers, Z stands for the set. When an is divisible by b and there is an integer c such that ac= b, we say that for every a and b in Z, a | b. In cases when the integer an is not divisible by b, we express it as 6 b|. (When a, b, and c are all yes, that's the most crucial case.) Nevertheless, what constitutes holds true regardless of whether any of them are negative or zero.) It may be easily seen that for each X and Y in $Z, a|(Xb + Yc)$ if $a|b$ and $a|c$.

To be *a* divisor of *b*, The form a | b is required for a to be positive. For an integer to be considered a nontrivial divisor of b, it must be a factor of b and be a member of the set 1{, b}. In mathematics, any integer greater than 1 is considered a prime number if does not split into any other positive integers other than itself and 1. Composite integers are positive integers larger than 1 that are not prime. Convention states proves that one is neither a composite or prime number.

Every positive integer up to ordering may be expressed distinctively in terms of prime products, according to a basic arithmetic theorem. That is, any integer that is positive $N > 1$ It also possible to express as N = $^{Q}{}_{i}$ $p^{e}{}_{i}i$, where the {p_i} separately prime and $e_i \geq 1$ for every one of us; furthermore, the {p_i} (and {e_i}) only known up to the point of placing an order.

Since we were elementary school students, we have been taught how to divide by the remainder. This idea is formalized by the following statement.

Proposition 1: Since We'll suppose that a and b are both positive integers. In reality, the numbers that follow are separate. in cases when the numbers q and r satisfy a=qb + r and $0 \leq r < b$.

The calculation of q and r for the given integers a and b may require polynomial time. which are the same as in the assertion. (The input length(s) ascertain the amount of time an algorithm. In conventional wisdom in algorithmic number theory is that binary is the appropriate representation for integer inputs. This part is really important. For this reason, the amount of time it takes for method that takes an integer N as its argument is measured by the length of its binary representation, kN k. as input. Keep in mind that $\|N\| = \lfloor \log N \rfloor + 1.$)

Two numbers the expressions for a and b are given by their $gcd(a,b)$ A positive integer c that is neither less than a nor greater than b must be found. No value is given for gcd (0,0). When a and b are negative, the GCD makes sense; nevertheless, for the vast majority of situations, we will find that $a,b \geq 1$; anyway, $gcd(a,b) = gcd(|a|,|b|)$.

Pay attention to the fact that $gcd(b,0) = gcd(0,b) = b$; Also, $gcd(a,p)$ has exactly two possible values: 1 or, if p is a prime number, then the inequality gcd(a,b) = 1 is satisfied. When a and b are prime numbers in absolute terms, we state that. Consider this realistic result:

Proposition 2: If a and b are substantially prime, we say that. Here is a practical outcome: at $Xa + Y\,b = gcd(a,b)$. Furthermore, $gcd(a,b)$ which, when written in this manner, is the lowest positive integer.

PROOF: Consider the set $I \stackrel{\text{def}}{=} \{\hat{X}a + \hat{Y}b \mid \hat{X},\hat{Y} \in \mathbb{Z}\}$ Keep in mind that $a,b \in I$, together with the certainty that I include a few positive numbers. Assume that d is the smallest positive in the set I. Our results demonstrate that $d = gcd(a,b)$ since d is also expressible as $^{d = Xa + Y\,b}$ If d is an element of I, the result is proved when X and Y are both elements of Z.

As evidence, we need to demonstrate that $^{d \mid a}$ and $^{d \mid b}$, and the biggest integer satisfying this condition is d. The reality is that Dividing all elements in I by d is a fact that can be proven. To see this, choose a random c from the set I and express it as c = X^0a + Y^0b, where X^0 and Y^0 are elements of the set Z. Dividing by the total yields the residual (Proposition 3.1) We own it, and its equation is c = qd + r, where q and r are integers and 0 r < d. After that

$$r = c - qd = X'a + Y'b - q(Xa + Yb) = (X' - qX)a + (Y' - qY)b \in I.$$

If $r \neq 0$, This goes against our decision to choose Because r is less than d, we may consider d to be which positive integer is the smallest inside the set I. For this reason, since r is equal to zero, d | c. Everything in I is subdivided by d, as this proves.

Data Analysis

Primes, Factoring, And RSA: We provide the earliest examples of what are often regarded as "hard" problems in number theory. One of the first issues will be discussed first: calculation of integer factors or just factoring.

Discovering a pair of positive integers the problem of factoring, where in this case, p and q are integers, and N is a composite number. that satisfy the equation pq= N. Given that factoring has long been known to be a challenging computing issue (even before its use in cryptography) and that it is very easy to explain makes it a prototypical example of a difficult problem. Solving the issue will take exponential time. $\mathcal{O}(\sqrt{N} \cdot \text{polylog}(N))$ via means of trial division, i.e., by thoroughly examining whether p divides N for $p = 2,\ldots,\lfloor\sqrt{N}\rfloor$. (This approach makes use of $\lfloor\sqrt{N}\rfloor$. polylogarithms, with each division $(N) = \|N\|^{c}$ period for a fixed c.) Since N/2 is the maximum size for a prime factor of N, this always works, even if N/2 is also the smallest possible. $\lfloor\sqrt{N}\rfloor$. Despite several years of research, no polynomial-time factoring algorithm has been proven, even if there exist algorithms with better running time.

Take this experiment with algorithm A and parameter n as an example.:

W-Factor, a weak factoring experiment$_{al}$(n):

Pick two consecutive numbers with n bits x_1, x_2.

To get N, divide x_1 by x_2.

When given N, A produces $\mathbf{x^0}_1, \mathbf{x^0}_2 > 1$.

The experiment's result is characterized as 1 if $x^0{}_1 \cdot x^0{}_2$ = N, and 0 in any case.

It is often assumed that the factoring issue is difficult, as we have just said. Are we to understand that

$$Pr[w - FactorA(n) = 1] \leq negl(n)$$

for all ppt algorithms is minimal A? Absolutely not. To begin, since x1 or x2 must be even for the number N to be even in the previous experiment (which it is with a 3/4 chance), A may easily factor N. We can make A's work harder by making it produce integers of length n, but it's still possible that small prime factors might have been readily discovered by x1 or x2 (and so by N). In order to avoid this in cryptography applications,

Generating Random Primes

Choosing n-bit numbers at random until we discover one that is prime is A natural method for producing a prime number with n bits; we may continue this process up to t times or until we succeed. A high-level explanation of the procedure may be found in Algorithm 4. 31.

Generating a random prime – high-level outline

Input: Length n; parameter t
Output: A uniform n-bit prime

for $i = 1$ to t:
  $p' \leftarrow \{0,1\}^{n-1}$
  $p := 1 \| p'$
  if p is prime **return** p
return fail

Be aware that by setting the advanced component of p to "1," the approach guarantees so, rather than a string of integers, the output will be a value-bound integer with a length of n. An "integer of length n" is defined in this book as an integer whose length, expressed in binary form, is 1 as the leading bit, is precisely n bits.

In the absence of a failure condition, the aforementioned procedure, when given an integer p, produces an n-bit uniform prime. Setting t so that we receive a failure probability that is insignificant inn is important for our purposes, since it determines the likelihood that the algorithm outputs will fail. By ensuring that the high-order bit of p is set to "1," the approach guarantees that the output will be an integer of length exactly n, rather than a value with boundaries, and so efficiently creates primes. However, in order to prove this, further information on two difficulties is required. In this context, a "integer of length n" is a numerical number whose primality of an integer p is such that it's the length of a binary representation is precisely n bits when the first bit is set to 1.

Theorem 3: According to as stated by Bertrand, each n-bit integer, the percentage of prime integers is at least 1/3n. positive integers n.

This means that, going back to the method for producing primes that was discussed before, if we set t = $3n^2$ Therefore there is a maximum chance that a prime number will be selected in all t algorithm iterations.

$$\left(1-\frac{1}{3n}\right)^{t} = \left(\left(1-\frac{1}{3n}\right)^{3n}\right)^{n} \leq \left(e^{-1}\right)^{n} = e^{-n}$$

in terms of n, which is insignificant when applying Inequality A.2). So, we may get an algorithm with a risk of output failure that is minimal in n by employing poly(n) iterations. (Practically, even fewer iterations are required since tighter findings than Theorem 4. are known.)

Testing primality: Finding the primness of an integer quickly is an ancient problem. Primality testing techniques were first created in the 1970s. These algorithms were based on probability and guaranteed that, given a prime integer p as input, the program would always return the word "prime." In contrast, the method would almost always return "composite" if p were composite, but it may provide the incorrect response ("prime") with a chance that is insignificant according to p. Say it again: if the procedure gives you "composite " as an output, then p is unquestionably composite; but, if "prime" is returned, then p is probably prime, but an error may have happened (and p is really composite). 2

***Primality Testing**

Next, we will prove and explain the Miller-Rabin primality test. None of the subsequent chapters make direct use of this content.

Discovering a feature that differentiates primes from composites is crucial to the Miller-Rabin algorithm. Enumerate The input value that has to be evaluated as N. First, we see that in the case of prime numbers, if N...$|\mathbb{Z}_N^*| = N-1$, hence, in the event that $a \in \{1,\ldots,N-1\}$ we have $a^{N-1} = 1$ given by Theorem mod N. This suggests that we may discover the primeness of N may be determined by choosing a uniform element a

and seeing whether the modulus of a^(N-1)? is 1. N is not a prime if and only if N-1≦1mod N. But in cases when N is not prime, we may continue to think that we will choose a rather often. By applying this test repeatedly, we may safely determine N's primeness if N-1≠1a mod N. The algorithm demonstrates the previously mentioned technique. Finding the exponentiation modulo N and the greatest common divisor are two things you should keep in mind at all times.

Picking choose a consistent material of $\{1,\ldots,N-1\}$ similarly completes in polynomial time.

Primality testing – first attempt

Input: Integer N and parameter 1^t
Output: A decision as to whether N is prime or composite

for $i = 1$ to t:
  $a \leftarrow \{1,\ldots,N-1\}$
  if $a^{N-1} \neq 1 \bmod N$ **return** "composite"
return "prime"

When for prime numbers, the procedure always gives the result "prime." "Composite" describes the outcome of the method in the event that it discovers N in any iteration if There is a polynomial N. If A is a modulo N element of the set {1,...,N -1}, then the condition that a^(N-1)≠1 holds. In order for N-1 to be divisible by 1, modulo N, it must be in the set Z_N^*. In the absence of a value smaller than 1, the equation [a^(N-1) mod N] cannot have a value of 1. Let us focus on a strictly belonging to Z_N^* for the moment. Assuming we are in the position of a witness who has independently confirmed that N is composite, we may deduce that, for every given a, a^(N-1) 6≠1 mod N. When the value of N is uncertain, all we can do is pray that the algorithm correctly identifies the witness with a "high" probability. is composite, because there are many witnesses. So long as there's a witness, this gut feeling is right. Two group-theoretic lemmas are required for the proof to be valid.

Proposition 2 Let $\mathbb{G}$ the set must be finite, with H being a subset of G. Assuming that H is true for all values of a and b

∈ H, we may deduce that ab ∈ H. As a result, H is a family inside G.

PROOF: First, it is essential that $\mathbb{H}$ is assumed to be within the scope of the group's operations, and it fulfills all the requirements. Relationships between $\mathbb{H}$ automatically passes down from $\mathbb{G}$. Let m = | $\mathbb{G}$ | (this is the point at which it is true that $\mathbb{G}$ is finite), when taking into account any given component a ∈ $\mathbb{H}$. Finishing up $\mathbb{H}$ indicates that$\mathbb{H}$ contains $a^{m-1} = a^{-1}$ along with $a^{m} = 1$. Thus, $\mathbb{H}$ includes not just the identity but also the inverse of every element.

Lemma 1: Let $\mathbb{H}$ constitute an exact subset of a limited set $\mathbb{G}$ (i.e., $\mathbb{H} \neq \mathbb{G}$). Then $|\mathbb{H}| \leq |\mathbb{G}|/2$.

PROOF: Let $\bar{h}$ play a role in $\mathbb{G}$ outside of $\mathbb{H}$; since $\mathbb{H} \neq \mathbb{G}$, Our knowledge includes such an $\bar{h}$ does exist. Think about the list $\bar{\mathbb{H}} \stackrel{\text{def}}{=} \{\bar{h}h \mid h \in \mathbb{H}\}$. Our results demonstrate that (1) $|\bar{\mathbb{H}}| = |\mathbb{H}|$, and (2) there is no element in $\bar{H}$ that is not outside of H, close to the point where H and $\bar{H}$ meet. Considering that each$\mathbb{H}$ and $\bar{\mathbb{H}}$ comprise parts of $\mathbb{G}$, they suggest $|\mathbb{G}| \geq |\mathbb{H}| + |\bar{\mathbb{H}}| = 2|\mathbb{H}|$, verifying the statement.

For any $h_1, h_2 \in \mathbb{H}$, if $\bar{h}h_1 = \bar{h}h_2$ the next day, by $\bar{h}^{-1}$ on both sides, there are $h_1 = h_2$. This demonstrates that each individual component h ∈ H is equivalent to a separate component $\bar{h}h \in \bar{\mathbb{H}}$, proving (1).

Go with the premise that $\bar{h}h \in \mathbb{H}$ within a certain amount of time. In other words, $\bar{h}h = h'$ for a few $h' \in H$, and so $\bar{h} = h'h - 1$. Now, $h'h - 1 \in H$ given that H is a small subset of $h', h - 1 \in H$. However, this implies that $\bar{h} \in H$, in direct opposition to how $\bar{h}$ ended up being selected. The lemma proof is now complete. and demonstrates (2).

In order to examine the previously provided algorithm, we may use the hypothesis that follows.

Methods for Constructing Pythagorean Triples

One way to get primitive triples that are both geometric and Pythagorean is to use the formula

$a = m^2 - n^2$
$b = 2mn$
$a = m^2 + n^2$

After that, an equilateral triangle formed by a, b, and c.

Example:
m = 3 and n = 2
$a = 3^2 - 2^2 = 9 - 4 = 5$
$c = 3^2 + 2^2 = 13$
$b = 2 \times 3 \times 2 = 12$

The outcome was the Pythagorean Triple (5,12, 13).

Similarly, Three, four, and five make up the Pythagorean Triple when m = 2 and n = 1.

Tree of Primitive Pythagorean Triples

The idea According to mathematical theory, a "tree of primitive Pythagorean triples" data structure may accommodate an endless number of distinct primitive Pythagorean triples by allowing each node to branch to several following nodes.

The set of all basic Pythagorean triples may be naturally described by a rooted tree, and a ternary tree in particular. The first finding of this was made by B. Berggren in 1934. Figure 2 shows the tree that represents the fundamental Pythagorean triples.

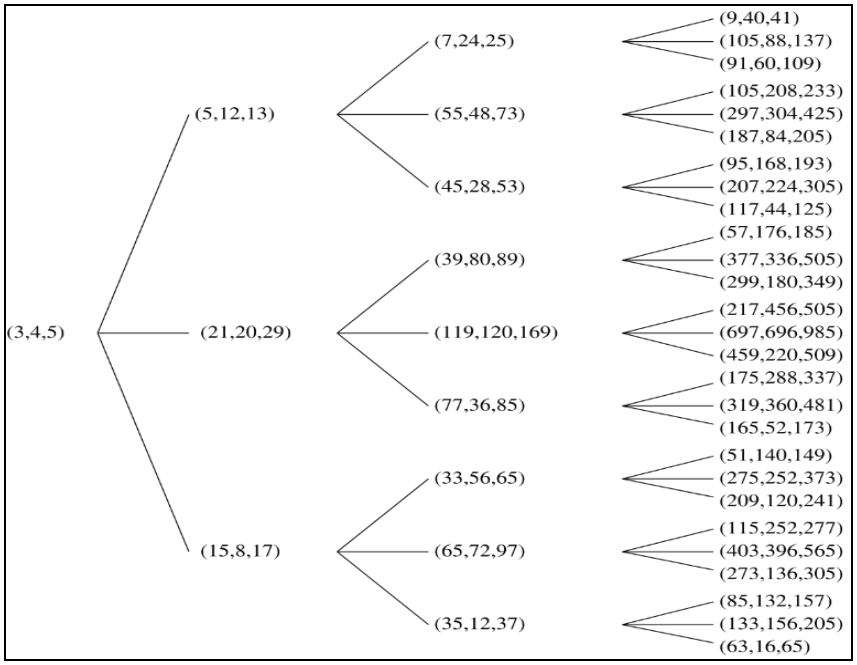

Fig 2: The arboreal structure of basic Pythagorean triples

When one of the three matrices is filled out, F. J. M. Barning hewed

$$A = \begin{pmatrix} 1 & -2 & 2 \\ 2 & -1 & 2 \\ 2 & -2 & 3 \end{pmatrix} \qquad B = \begin{pmatrix} 1 & 2 & 2 \\ 2 & 1 & 2 \\ 2 & 2 & 3 \end{pmatrix} \qquad A = \begin{pmatrix} -1 & 2 & 2 \\ -2 & 1 & 2 \\ -2 & 2 & 3 \end{pmatrix}$$

when it's done with a column vector of Pythagorean triples on the right side, it generates an additional column vector that contains different Pythagorean triples. All future triples will be primitive if the originating triple is primitive. So, there are three "offspring" for every basic Pythagorean triple. No primordial triple exists more than once; all primitive Pythagorean triples are therefore derived from the triple (3,4,5). An infinite ternary tree with the node at (3,4,5) may be used to visually display the outcome. A. R. Kanga's 1990 study and A. Hall's 1970 paper both included this tree.

Parallelepiped

The three-dimensional shape made up of six parallelograms is called a parallelepiped or rhomboid. Similar to how a cube is connected to a square or a cuboid to a rectangle, it is related to a parallelogram. The four geometrical ideas in Euclidean geometry-the parallelepiped, the parallelogram, the cube, and the square-are all included in its definition. From this point of view, the only valid definitions of Affine geometry that take into account undifferentiated angles are parallelograms and parallelepipeds. As shown in Figure 3, the parallelepiped.

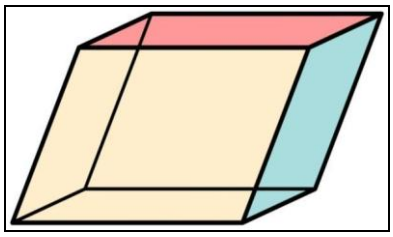

Fig 3: Parallelepiped

Another way to define a parallelepiped is as:

- An equilateral polyhedron with six faces, called a parallelogram
- Three sets of parallel faces form a hexahedron, and
- A prism with a parallelogram for its base.

Examples of parallelepiped in particular include:

- Cuboid with six square profiles
- A cube with six equal sides, and
- Rhombohedron, with six rhombus faces combined.

Properties of Parallelepiped

Anyone of the three sets of parallel sides might serve as the prism's basis plane. The edges of a parallelepiped are all the same length and form three sets of four. Linear transformations of a cube give rise to parallelepiped.

Special Cases of Parallelepiped

Rectangular Parallelepiped

All of the faces of a rectangular cuboid are rectangular, making it a kind of parallelepiped. All of a cube's faces are square, making it a special kind of cuboid.

Perfect Parallelepiped

Optimal parallelepipeds have face diagonals, space diagonals, and edges of integer length. It is possible to find certain complete parallelepipeds with two rectangular sides. The existence of any parallelepiped with just rectangular faces is, however, unknown at this time. A perfect cuboid would be the name given to this kind of Parallelepiped if this is the case.

Transplantation of Primitive Pythagorean Tree

Use the parallelepiped's edges as the basic Pythagoreans and create the primitive vertex at the furthest point. In order to construct a cuboid, the given coordinates-a, 0 at 0, 0 at b, 0, 0 at c-must be entered as follows: (a at b, 0), 0 at b, c, (a at 0 with c), AND (a at b with c). use the parallel planes to connect to the coordinate planes. The basic diagonal is c√2. Building a tree like this is known as transplanting a rudimentary Pythagorean tree.

Figure 4. shows the transplanting of a basic Pythagorean tree using several primitive Pythagoreans.

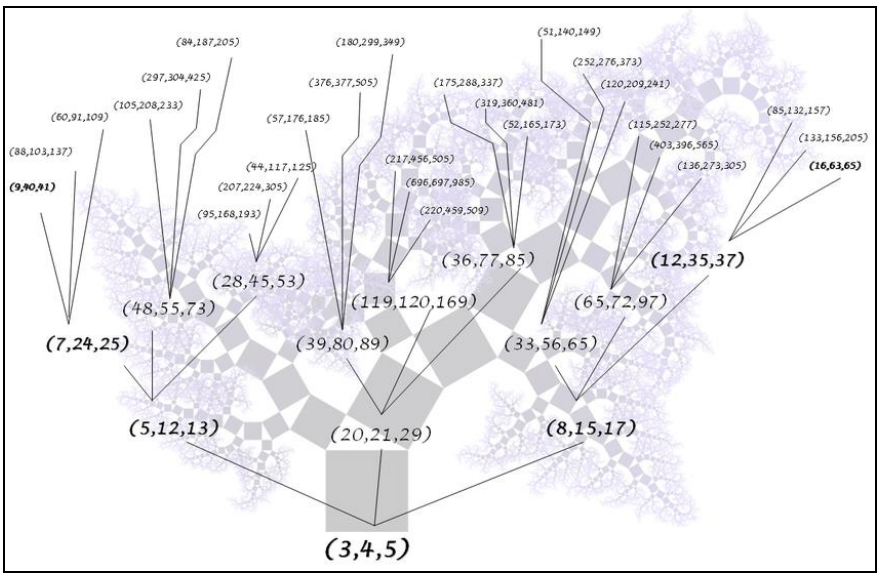

Fig 4: Transplantation of primitive Pythagorean tree

Theorem 4: Two Pythagorean trees are always distinct.

Proof: Let us consider T_1 and T_2 be the two Pythagorean trees. Assume (a, b, c) and (p, q, r) to represent the Pythagorean primitives of T_1 and T_2 respectively

Since, (a, b, c) and (p, q, r) are Pythagorean primitives, therefore we have

$a^2 + b^2 = c^2$ and $p^2 + q^2 = r^2$

where, $a < b$ and $p < q$.

Let, if possible, $c^2 = r^2$

therefore, $a^2 + b^2 = p^2 + q^2$

$a^2 - p^2 = q^2 - b^2$

$$(a - p)(a + p) = (q - b)(q + b)$$

then, either $(a - p) = (q - b)$ and $(a + p) = (q + b)$

or $(a - p) = (q + b)$ and $(a + p) = (q - b)$

if $(a - p) = (q - b)$ and $(a + p) = (q + b)$

then, $a = q$ and $p = b$ which gives a > b, which is a contradiction.

and if $(a - p) = (q + b)$ and $(a + p) = (q - b)$

and because a, b, c, p, q, and r are all unchangeable, we get a = q and p = - b. every single one of them is optimistic

Therefore, no two Pythagorean primitives can ever be identical. As a result, no two Pythagorean trees built with Pythagorean primitives are ever the same.

Conclusion

In computational number theory, finding primality requires a battery of tests and procedures, the most important of which are the Miller-Rabin and AKS tests, which guarantee answers for primality testing in polynomial time. Goldwasser and Kilian's elliptic curve test is a notable example of a technique; subsequent versions, such as the ECPP variation, have improved upon it. Adleman, Pomerance, and Rumley's APR test and Pollard's p−1 approach both add to the primality testing landscape. Techniques like Shanks' Step and Giant-Step approaches have been the subject of substantial effort on discrete logarithm problems (DLP), and index calculus methods like NFS and FFS are essential for efficient solution of DLP. With the proliferation of the Internet, cryptography's importance in ensuring the safety of data transmission has skyrocketed, making it imperative that users have a solid grasp of cryptographic concepts.

References

1. Hamlin N. Number in mathematical cryptography. Open Journal of Discrete Mathematics. 2017;7:13–31. doi:10.4236/ojdm.2017.71003.
2. Saranya C, Janaki G. Some interesting applications of number theory. 2021.
3. Annu. Algebraic structures and their applications in modern cryptography. Innovative Research Thoughts. 2024;10:52–59. doi:10.36676/irt.v10.i3.1433.
4. Asole R, Gaikwad S. Applications of group theory in cryptography. Gurukul International Multidisciplinary Research Journal. 2025. doi:10.69758/GIMRJ/2504I5VXIIIP0036.
5. Easttom W. Modern cryptography: applied mathematics for encryption and information security. Cham: Springer; c2021. doi:10.1007/978-3-030-63115-4.
6. Dorostkar Z. Mathematics for cryptography: a guide to mathematical fundamentals of different classes of cryptography algorithms; c2023.
7. Qian Y. Application of modern algebra in cryptography. Theoretical and Natural Science. 2023;10:16–20. doi:10.54254/2753-8818/10/20230304.
8. Srungaram V. Cryptographic protocol. International Journal of Mathematics and Its Applications; c2016.
9. John M, Ogoegbulem O. Number theory in RSA encryption systems. International Journal of Mathematics. 2023;6:7–16. doi:10.5281/zenodo.10209900.
10. Wang T, Xu Z. The application of group theory behind modern cryptography. Theoretical and Natural Science. 2023;13:195–201. doi:10.54254/2753-8818/13/20240844.