Journal of *Cybersecurity and Privacy*

MDPI

Article

Continued Fractions Applied to the One Line Factoring Algorithm for Breaking RSA

Anthony Overmars and Sitalakshmi Venkatraman *

Department of Information Technology, Melbourne Polytechnic, 77 St Georges Rd, Preston, VIC 3072, Australia; anthonyovermars@melbournepolytechnic.edu.au

* Correspondence: sitavenkat@melbournepolytechnic.edu.au

Abstract: The RSA (Rivest–Shamir–Adleman) cryptosystem is an asymmetric public key cryptosystem popular for its use in encryptions and digital signatures. However, the Wiener's attack on the RSA cryptosystem utilizes continued fractions, which has generated much interest in developing competitive factoring algorithms. A general-purpose integer factorization method first proposed by Lehmer and Powers formed the basis of the well-known Continued Fraction Factorization (CFRAC) method. Recent work on the one line factoring algorithm by Hart and its connection with Lehman's factoring method have motivated this paper. The emphasis of this paper is to explore the representations of $\frac{P}{Q}$ as continued fractions and the suitability of lower ordered convergences as representations of $\frac{a}{b}$. These simpler convergences are then prescribed to Hart's one line factoring algorithm. As an illustration, we demonstrate the working of our approach with two numbers: one smaller number and another larger number occupying 95 bits. Using our method, the fourth convergence finds the factors as the solution for the smaller number, while the eleventh convergence finds the factors for the larger number. The security of the RSA public key cryptosystem relies on the computational difficulty of factoring large integers. Among the challenges in breaking RSA semi-primes, RSA250, which is an 829-bit semi-prime, continues to hold a research record. In this paper, we apply our method to factorize RSA250 and present the practical implementation of our algorithm. Our approach's theoretical and experimental findings demonstrate the reduction of the search space and a faster solution to the semi-prime factorization problem, resulting in key contributions and practical implications. We identify further research to extend our approach by exploring limitations and additional considerations such as the difference of squares method, paving the way for further research in this direction.

Keywords: continued fractions; one line factoring; prime factorization; semi-primes; Wiener's attack; public key cryptography; RSA attack; cybersecurity

Citation: Overmars, A.; Venkatraman, S. Continued Fractions Applied to the One Line Factoring Algorithm for Breaking RSA. *J. Cybersecur. Priv.* **2024**, *4*, 41–54. https://doi.org/10.3390/jcp4010003

Received: 13 October 2023
Revised: 31 December 2023
Accepted: 5 January 2024
Published: 10 January 2024

1. Introduction

The RSA (Rivest–Shamir–Adleman) cryptosystems apply large semi-primes in an encryption algorithm for the generation of public keys and private keys to successfully secure an online communication [1–3]. The security of such cryptosystems depends on prime factorization of large numbers being a huge computing challenge [4–6]. Though the semi-prime factorization of large numbers has always interested mathematicians for centuries, the practical importance of preserving the security of these cryptosystems has led to exploration of the limitations of existing approaches towards the emergence of new methods, thereby redefining the cybersecurity landscape from time to time [7–13].

In the past two decades, integer factorization algorithms have evolved by improving existing ones to a great extent such that very large semi-primes of more than 250 decimal digits can be factorized with sufficient computing power [14–17]. Some of the famous integer factorization algorithms are Lenstra's elliptic curve algorithm [18], Pomerance's quadratic sieve, and the General Number Field Sieve [19,20]. Not only mathematicians,

but also computing professionals and organizations have been exploring the properties of primes for several decades. More recently, their research directions have moved towards developing efficient tests for primality with the key objective of creating several applications in the domain of information security, particularly the RSA public key cryptosystem. A sieve in this context refers to the process of searching for prime numbers among all the integers up to a prescribed bound [6,20,21]. The early practical building blocks of public-key cryptography are attributed to the Diffie–Hellman protocol over finite fields and the RSA cryptosystem. Though several other cryptographic primitives have entered the landscape, the RSA and finite field Diffie–Hellman methods are more prevalent in terms of their use for secure information transmission and key exchange in the computing field [22]. The key sizes in these primitives are much longer when compared to elliptic curve-based cryptosystems. Thus, finding semi-primes has been a challenge and has drawn immense attention among researchers looking to break the RSA cryptosystem [23,24].

Our previous research work involved proposing several factorization methods including approaches using Pythagorean triples, sum of squares, and polynomials, as well as investigation of their impact on RSA attacks [25–30]. Readers are encouraged to refer to our previous work and related published articles in the literature that provide the preliminary definitions and commonly used mathematical notations adopted in this work. In this paper, we take a modest step to further the method of continued fractions by applying one line factoring algorithms with an effect on semi-prime factorization of the RSA primitive. The main aim of this paper is to characterize k:kN as per Lehman [31] and apply this to Hart's one line factorization method [32]. Our research work is the first of its kind in exploring the relationship between the convergence of PQ and the attributes of k, which is used in Hart's one line factorization algorithm and has applications in cryptography. Little is known about the attributes of k, and our research provides a better understanding of its relationship with the convergence of a solution. This paper presents the theory of our factorization approach and illustrates the working of our method with examples for breaking keys of different sizes. Our key contribution lies in the reduction of the search space and the faster solution to the semi-prime factorization problem. We provide the practical implementation and results of our experimentation conducted using small and large semi-primes as well as RSA250, which has 250 decimal digits (829 bits). In addition, our modest contribution includes considering a general form of the Brahmagupta–Fibonacci identity [33] and applying it to the difference of squares, which extends our earlier work on the sum of squares approach for semi-prime factorization [15,17].

The organization of this paper is as follows: Section 2 provides a brief background history and theory on continued fractions. In Section 3, we present our method to factorize using continued fractions as a worked example. The application of our approach for factoring RSA250 is detailed in Section 4. Our practical observations are described in Section 5. Section 6 provides suitable measures of performance for our proposed method and future research ideas. Section 7 describes the difference of squares and, finally, in Section 8, the conclusions and contributions of this ongoing research are summarized.

2. Theory

The theory on continued fractions and the history of various factorization methods form the back backbone of this research investigation. Fermat's factorization method which was formulated in the early 1600's uses the difference of squares and was published much later [34–36]. However, most modern factorization algorithms are based on Fermat's factorization method [37–40]. In the 1920s, Maurice Kraitchik presented a notable improvement to Fermat's factorization. Instead of having a number n as a difference of squares, Kraitchik determined that it would suffice to find a difference of squares equal to a multiple of N [41]. This can be mathematically stated as $X^2 \equiv Y^2 \ mod \ N$. In 1985, Pomerance's Quadratic Factoring Algorithm [19] used Kraitchik's scheme with the steps given below:

(1) Generate congruences $U \equiv V \ mod \ N$.
(2) Determine factorizations of U and V for some of the congruences.

(3) Determine a subset of the factored congruences which, when multiplied, produce a special congruence $X^2 \equiv Y^2 \; mod \; N$.
(4) Factorization of N : $\gcd(X - Y, N)$.

Consider $X^2 \equiv Y^2 \; mod \; N$. This can be rewritten as given below:

(1) $X^2 - Y^2 \equiv 0 \; mod \; N$.
(2) $X^2 - Y^2 = kN$.
(3) $(X - Y)(X + Y) = kN$.
(4) $P = \gcd(X - Y, kN)$ or $Q = \gcd(X + Y, kN)$.

Our premise is that if we can find such a solution as mathematically derived in the above steps, then the greatest common divisor (gcd) will be a solution to the factorization of N : $N = PQ$. We carefully consider Knuth's work of 1981, which states that attempting to factorize a semi-prime $N = PQ$ with $4kN$ "is a rather curious way to proceed, if not down right stupid" [42]. Further, according to Knuth, finding k is clearly a chicken and egg scenario. In this work, we aim to perform experiments to investigate the characteristics of k by working backwards from the answer and narrowing the search field for possible k values. Further, we study many correct solutions which are the convergences of PQ, which have not been explored previously. We believed that this ongoing research work would lead to interesting findings and make better use of our narrower definition of the parameterization of k.

We next provide a brief background of related articles and their evolution in this domain. Continued Fraction Factorization (CFRAC) is well defined among factorization algorithms and is the baseline against which all others are measured. In number theory, the CFRAC method was the first general-purpose factorization algorithm that was efficiently implemented in 1975 as a computer algorithm by Morrison and Brillhart [7] using the continued fraction approach of $\sqrt{kN}$. A few years later, Pomerance and Wagstaff explored this further and considered in detail the Knuth–Schroeppel function, which attempted to find suitable values for k that generated new ideas for factoring large integers [43,44]. Subsequently, Williams and Shallit provided a computational history of factoring and the refinement of several ideas in this space [45]. While such combinations of number theory ideas have their mathematical merits, in this paper, our interest lies in finding a more simple and elegant approach that can motivate many researchers from the information security domain to explore further in this direction. We focus on the initial work published in 1974 by Lehman, who considered the ratio of $\frac{P}{Q}$ as $\frac{a}{b}$ such that $x^2 - y^2 = 4kN, k = ab$ [46]. After nearly four decades, Hart considered Lehman's work and ideas from Shank's method (SQUFOF) reported in 1982 [31] and subsequently developed a one line factoring algorithm in 2012 [32].

From the above equation provided in (2), namely $X^2 - Y^2 = kN$, we obtain $Y^2 = X^2 - kN$. If we let $X = \left\lceil \sqrt{kN} \right\rceil$, then $Y^2 = \left(\left\lceil \sqrt{kN} \right\rceil \right)^2 - kN$.

Our simple approach looks for a k value which provides a perfect square. The greatest common divisor $(X - Y, N)$ will be a solution to the factorization of N : $N = PQ$.

We propose a new method that considers the continued fraction of the factorization of N. The mathematical representation and the logic of the algorithm are as follows.

$P/Q \approx p/q$, where p/q are the convergences of P/Q.

$$Y^2 = \left(\left\lceil \sqrt{kN} \right\rceil \right)^2 - kN \; \Rightarrow \; Y^2 = \left(\left\lceil \sqrt{4pqN} \right\rceil \right)^2 - 4pqN$$

The circular conundrum observed by Knuth [31] is that $k = 4pq$ and that $\frac{p}{q} \approx \frac{P}{Q}$.

If P/Q is known, then a convergence $\frac{p}{q}$ can be found and then a suitable k value can be determined. Hence, we have the chicken and the egg conundrum:

(1) We cannot know $\frac{p}{q}$ because we do not know $\frac{P}{Q}$;
(2) We cannot solve for $\frac{P}{Q}$ because we do not know $\frac{p}{q}$.

However, our study in this direction is not limited by Knuth's observation that this "is a rather curious way to proceed, if not Down Right Stupid" [42]. Regardless of the obvious short comings provided by Knuth, we explore our proposed approach with worked examples in the next section, which provides the basis of our main investigation into applying our method for successfully factorizing RSA250 in the subsequent section.

3. Worked Examples

We provide the verified working of our factorization approach using two worked examples in this section. In the first example, we consider an integer (13,290,059) to factorize using continued fractions. This can be achieved with a brute force search by incrementing through k until $k = 4pq \Rightarrow p/q = 11/8$. This provides us with a factorization of 13,290,059 with factors $P = 4261$ and$Q = 3119$.

Let us now consider the solution and describe the ratio of P/Q as a continued fraction such that 4261/3119 => 1 2 1 2 1 2 1 1 3 12.

We will now consider each convergent fraction of the continued fraction in turn until we find a k value which provides a solution to $x^2 - y^2 = kN, k = 4pq$.

Using our approach, the fourth convergence for 4261/3119 => 1 2 1 2 1 2 1 1 3 12 provides the solution. The first four convergences are indicated by [k,x,y,z]. Note that when $z = 0$, a solution is found.

1. $\frac{p}{q} = \frac{1}{1} \Rightarrow [k,\ x,\ y,\ z] = [4,\ 7292,\ 114,\ 5]$.
2. $\frac{p}{q} = \frac{3}{2} \Rightarrow [k,\ x,\ y,\ z] = [24,\ 17860,\ 134,\ 15]$.
3. $\frac{p}{q} = \frac{4}{3} \Rightarrow [k,\ x,\ y,\ z] = [48,\ 25258,\ 209,\ 7]$.
4. $\frac{p}{q} = \frac{11}{8} \Rightarrow [k,\ x,\ y,\ z] = [352,\ 68397,\ 221,\ 0]$.

For the example considered here, we arrive at $gcd(68397 - 221, 13290059) = 4261$. From the above, it can be verified easily that the fourth convergence finds the factors as the solution for this example. We note that all other convergences after this one also provide suitable values for k that lead to a solution. Other approaches such as Silverman's method require many polynomials to generate the residues [47].

As a second example, we consider the integer 21,565,941,721,999,797,939,843,713,963, a 95-bit prime number as illustrated by Crandall and Pomerance [48] on pg307. To factorize using continued fractions, the solution can be achieved with a brute force search by incrementing through k until the following value is reached:

$$k = 4pq \quad \Rightarrow \quad p/q = 17887/6172$$

This obtains a factorization of 21,565,941,721,999,797,939,843,713,963 with factors $P = 1000000000000037$ and $Q = 21565941721999$.

Consider the solution describing the ratio of P/Q as a continued fraction such that 1,000,000,000,000,037/21,565,941,721,999 $\Rightarrow$ 46 2 1 2 2 2 2 1 1 1 3 8 13 1 263 1 5 38 1 4 1 2 8 2 1 1 2.

The continued fraction length is 26 and, by applying our approach, the 11th convergence provides a suitable k value and the resulting factorization shown below:

1. $\frac{p}{q} = \frac{46}{1} \Rightarrow [k,\ x,\ y,\ z] = [184,\ 1992017388691164,\ 33696672,\ 4942]$.
2. $\frac{p}{q} = \frac{93}{2} \Rightarrow [k,\ x,\ y,\ z] = [744,\ 4005628619975628,\ 66115834,\ 9235]$.
3. $\frac{p}{q} = \frac{139}{3} \Rightarrow [k,\ x,\ y,\ z] = [1668,\ 5997665445179121,\ 26939199,\ 4084]$.
4. $\frac{p}{q} = \frac{371}{8} \Rightarrow [k,\ x,\ y,\ z] = [11872,\ 16000964349800346,\ 91024971,\ 12000]$.
5. $\frac{p}{q} = \frac{881}{19} \Rightarrow [k,\ x,\ y,\ z] = [66956,\ 37999594654919919,\ 84176579,\ 11777]$.
6. $\frac{p}{q} = \frac{2133}{46} \Rightarrow [k,\ x,\ y,\ z] = [392472,\ 92000153692897192,\ 370945991,\ 18331]$.
7. $\frac{p}{q} = \frac{5147}{111} \Rightarrow [k,\ x,\ y,\ z] = [2285268,\ 221999902043111349,\ 519272009,\ 23683]$.
8. $\frac{p}{q} = \frac{7280}{157} \Rightarrow [k,\ x,\ y,\ z] = [4571840,\ 314000055736153583,\ 655378625,\ 20303]$.
9. $\frac{p}{q} = \frac{12427}{268} \Rightarrow [k,\ x,\ y,\ z] = [13321744,\ 535999957779289827,\ 962310841,\ 18309]$.

10. $\frac{p}{q} = \frac{19707}{425} \Rightarrow [k, x, y, z] = [33501900, 850000013515449911, 875520517, 15650]$.
11. $\frac{p}{q} = \frac{71548}{1543} \Rightarrow [k, x, y, z] = [441594256, 3085999998325641543, 1674472639, 0]$.

For the example considered here, we arrive at $gcd\,(3085999998325641543 - 1674472639, 215659417219997979398437139 63) = 21565941721999$.

From the above, it can be verified that the eleventh convergence finds the factors as the solution for this example. As per the previous example, all other convergences after this one also provide suitable values for k and lead to a solution. The two worked examples shown above demonstrate the simplicity of our proposed approach and the steps for convergence of a solution. The first example is a smaller integer, while the second is a very large integer, which is used for illustration in other studies found in the literature [48]. From the above-described worked examples, we arrive at the following three general properties:

- $gcd(p, q) = 1$.
- $\exists \frac{p}{q} \in \mathbb{R}^+ : \frac{p}{q} \approx \frac{P}{Q}, pq \ll PQ$.
- $\frac{P}{Q}$ is unknown.

Our main interest is in applying our approach to the breaking of keys as key sizes for RSA and finite field Diffie–Hellman have become unwieldy. In the next section, we demonstrate the use of our approach to factorize RSA250.

4. Application to Breaking RSA250

At the center of secure information transmission is the RSA cryptosystem. An information transmission scheme that meets a 128-bit security strength is widely accepted with a key size of approximately 3072 bits [24]. Even though high computing power is currently affordable with advancements in information and communication technologies, there are certain practical reasons for public key cryptographic schemes employing weak keys. One main reason for implementing weak key sizes is to ensure backward compatibility. Further, small key sizes are used in embedded environments with a scarcity of computational power for public-key cryptographic operations. Hence, several research studies assess the difficulty of the mathematical problem of prime factorization that underpins the security of RSA. They focus on the feasibility of integer factorization and the RSA factoring challenges thereof. A recent research study's target was the RSA240 factoring challenge [24], while previous records were related to breaking 768-bit keys [23,28]. RSA250 has 250 decimal digits (829 bits) and was only factored in 2020, which makes it a good candidate for exploring our method as it is of great research interest in the field of information security.

In this paper, let us now consider RSA250 to apply our approach of factorization where we limit the search using the attributes P and Q. We note that $\frac{P}{Q}$ is known for RSA250 and hence the convergences for $\frac{p}{q}$ are known. Let us consider the numbers given below for RSA250:

[N] 21403246502407449612644230728393335630086147151447550177977549208814180234471401366433455190958046796109928518724709145876873962619215573630474547705208051190564931066876915900197594056934574522305893259766974716817380693648946998715784949759374979 37

[P] 64135289477071580278790190170577389084825014742943447208116859632024532344630238623598752668347708737661925585694639798853367

[Q] 33372027594978156556226010605355114227940760344767554666784520987023841729210037080257448673296881877565718986258036932062711

The ratio of P/Q is given by the continued fraction below and has a length of 261. The italicized and underlined part of the continued fraction provides the first suitable convergence for p/q which leads to a solution.

1 1 11 1 3 1 4 2 1 2 6 3 1 1 1 1 1 1 26 1 4 1 4 4 7 1 3 1 1 2 3 2 1 1 7 1 4 1 1 1 5 1 1 1 1 6 12 5 1 1 2 3 1 2 1 1 2 77 1 32 21 1 1 3 5 1 2 11 3 1 1 1 30 1 4 4 1 2 5 2 1 2 1 21 1 2 3 1 7 2 1 1 4 9 1 25 1 28 1 1 1 34 1 1 4 1 2 1 6 2 6 5 5 2 4 1 3 3 2 5 1 1 2 3 1 1 3 5 1 1 5 1 1 3 2 36 2 3 2 1 1 1 9 12 1 1 3 1 1 4 1 4 1 14 1 2 2 6 1 1 1 2 25 47 1 1 5 1 5 1 2 3 2 1 183 1 1 1 1 4 3 1 3 1 1 2 3 1 6 19 1 1 2 1 2 2 1

1 1 4 4 1 44 3 1 29 1 12 3 17 1 1 1 11 1 36 36 2 1 8 14 1 7 5 2 1 3 1 8 16 19 1 6 9 1 3 2 1 1 1 2 1 1 4 2 2 1 221 1 1 4 1 3 1 1 4 2 2 17 1 4 2

The 93rd convergence produces a suitable k value which provides the solution for the factorization of RSA250 as given below:

$[k, x, y, z] =$

[2044342121097039395007571435252629683261030101582255392275672824550073295931132311 2,

661479843631644117228284492349830814822621363186543464872719039901780748603913907394254563141547525634166624762339395539405514099746668217426404901716820003797839993 5,

635656125074327398682139552816030653447112548284958974076574038453054287712216680 91,

0].

In the case of RSA250, the underlined part of the continued fraction convergence given above for the convergence of the ratio P/Q illustrates that, using our method, a convergence using only 93 of the 261 terms produces a suitable k value.

k = 2^3 × 3 × 7 × 103 × 281 × 3407 × 72,907 × 157,733 × 334,993 × 26,532,262,619,089 × 106,061,927,979,829 × 113,832,765,910,030,508,110,732,786,883.

In RSA250, the representations of P and Q have the same number of decimal digits. Using this property and the value of $\sqrt{N}$, a lower bound and an upper bound for possible k values can be determined [49]. The challenge is to find such a restricted search space for suitable k values. Using our method, an upper bound as well as a lower bound for k can be determined to limit that search area for suitable k values. For the case of RSA250, the ratio of P/Q convergences reduces to a problem of a ratio given by a 41-bit convergence divided by another 41-bit convergence that results in an upper bound and lower bound for k. The essence of this is that our method reduces the problem of the search space in a prime field with 125 decimal digits for RSA250, which has 250 decimal digits. The problem reduces to searching for ratios (convergences) whose denominators and numerators are 41-decimal digit composites. We note that one of the composites is 113,832,765,910,030,508,110,732,786,883, which is a 30-decimal digit number.

The purpose of choosing a known RSA250 factorization to apply our method is twofold: firstly, we are able to determine that a solution exists, and secondly, we are able to determine the practical implication of our approach in reducing the size of the numbers that need to be manipulated to arrive at such a semi-prime factorization solution faster for breaking keys in real-life applications. The authors do acknowledge that the search now involves a larger range of composite candidates for the numerator and the denominator in determining the lower bound and upper bound for suitable k values. Next, we provide further observations through the practical implementation presented that would interest researchers and practitioners in the field of information security.

5. Practical Implementation and Observations

In this section, we provide a practical implementation of our factorization approach using continued fractions and discuss our observations based on our experiment conducted with large integers. We note that brute force searching is feasible only for smaller integers. As reported in the literature, brute force methods are usually time-consuming and hence may not be efficient for many real-time applications [50,51]. Also, for large numbers, the k method may not be optimal. However, for all RSA numbers whose factorizations are known, a key observation provided in this paper shows that there is a relationship between the length of the continued fraction and the position where the first suitable k value appears. Further, all congruencies after this first k value are also valid k solutions. Exploring this relationship significantly reduces the search field for $k = 4pq$. More experimental exploration in this direction is an area of our ongoing research.

We provide a three-step algorithm below and the practical implementation along with the source code in the Java programming language. The Java code illustrates the

simplicity of the practical implementation of our algorithm and the potential to further develop applications in information security. Firstly, we present how the continued fraction is generated using the ratio of a known $\frac{P}{Q}$. We then use this set of convergences to create each convergence p/q. From this we then determine $k: \ k = 4pq$. We recall $Y^2 = \left(\left\lceil \sqrt{4pqN} \right\rceil\right)^2 - 4pqN$. An explanation of the three steps in the algorithm and the Java code of the implementation are provided below.

Step 1: Obtain the continued fraction using the factorization of $N: \ N = PQ$. An implementation of this step in the Java programming language was developed to practically demonstrate our simple approach. The continued fraction is returned as shown in the code snippet of our implementation in Java:

```
public static BigInteger [] ContinuedFraction(BigInteger [] R) {
      BigInteger [] ConFrac = new BigInteger [1000];
      BigInteger NUMER=R[0], DENOM=R[1];
      int i=0;
      while(!(DENOM.equals(BigInteger.ONE))) {
            R=NUMER.divideAndRemainder(DENOM);
            ConFrac[i]=R[0];
            NUMER=DENOM;
            DENOM=R[1];i++;
      }
      ConFrac[i]=NUMER;
      return ConFrac;
}
```

Step 2: For a specified number of convergence terms, calculate the convergence. The implementation in Java was developed as follows and the convergence fraction is returned.

```
public static BigInteger [] GetRatio(BigInteger [] ContinuedFraction,int SeriesLength) {
      BigInteger [] RATIO = new BigInteger [2];
      BigInteger RATIO2, NUMERATOR=BigInteger.ONE, DENOMINATOR;
      RATIO[1]=BigInteger.ONE; DENOMINATOR=ContinuedFraction[SeriesLength];
      for(int i=SeriesLength;i>0;i--) {
            RATIO2=DENOMINATOR;
            RATIO[1]=NUMERATOR;
            RATIO[0]=ContinuedFraction[i-1];
            NUMERATOR=RATIO2;
            DENOMINATOR=RATIO2.multiply(RATIO[0]).add(RATIO[1]);
      }
      RATIO[0]=DENOMINATOR;
      RATIO[1]=NUMERATOR;
      return RATIO;
}
```

Step 3: Test the convergence and see if this is a solution for which $k: k = 4pq$ that solves $Y^2 = \left(\left\lceil \sqrt{4pqN} \right\rceil\right)^2 - 4pqN$. The Java implementation of the logic is provided below and the result is presented as $[k, x, y, z]$.

```
Public static Boolean ConFracSeries(BigInteger [] ContinuedFraction,int SeriesLength,
      BigInteger N) {
      BigInteger [] RATIO = new BigInteger [2], KABC = new BigInteger [4];
      Boolean done=true;
      int Maximum_Factors = 1000;
      BigInteger [] R1Factors = new BigInteger [Maximum_Factors];
      BigInteger [] R2Factors = new BigInteger [Maximum_Factors];
      BigInteger K4;
      for(int i=0;i<SeriesLength;i++) {
            RATIO=GetRatio(ContinuedFraction,i);
            K4=RATIO[0].multiply(RATIO[1]).multiply(new BigInteger("4"));
            String str=""+RATIO[0]+"/"+RATIO[1]+"=>";System.out.print(str);
            KABC=Get_KABC(K4,N); str=" [k,a,b,c]=["+KABC[0]+","+KABC[1]+","+KABC[2]+
                  ","+KABC[3]+"]"; System.out.println(str);
            if(KABC[3].equals(BigInteger.ZERO)) {i=SeriesLength;}
      return done;
}
```

The astute reader will observe that for $[k, x, y, z]$, a solution is found when $z = 0$. Another observation is that there are intermediate solutions which may be used in concert to find a solution. Some ideas from Silverman using multiple polynomials may produce results [47]. Similarly, some recent research focusing on quantum computing and blockchain technologies requires that RSA cryptography be revisited [52–54]. However, we observe that it is still not easy to solve the RSA modulus using quantum computing. Shor's algorithm requires as many qubits as there are bits in the modulus [12,36,55]. Quantum computers which can attack RSA easily are not expected in the immediate future. These ideas form areas of future ongoing research and are beyond the scope of this paper.

6. Extension to Research Using Difference of Squares

Our research investigations through the experimental study conducted have resulted in several findings, as described in previous sections. These form encouraging foundations for us to explore the theory of the difference of squares as a future research direction. Our observations from the practical examples given above motivate us to extend our research using difference of squares. We now explore this idea of difference of squares with the Brahmagupta–Fibonacci identity [33], normally applied to P and Q each as a sum of two squares. We apply the Brahmagupta–Fibonacci identity to P and Q which we now express as a difference of two squares. We note that the Brahmagupta–Fibonacci identity has an inherent constraint in that it can only be applied to semi-primes congruent to $1 mod 4$. Further, the two primes forming the construction of the semi-prime must also be congruent to $1 mod 4$. This constraint limits the solvable semi-primes to a quarter of the possibilities. We are limited to only $1 mod 4 = 1 mod 4 * 1 mod 4$ for the solvable semi-primes as $3 mod 4 = 1 mod 4 * 3 mod 4$, $3 mod 4 = 3 mod * 1 mod 4$, $1 mod 4 = 3 mod 4 * 3 mod 4$ cannot be resolved.

We note that the Brahmagupta–Fibonacci two-square identity is analogous to Euler's four-square identity and this is relevant to the semi-prime factorization problem. A summary follows of the Brahmagupta–Fibonacci identity as well as its extensibility, which is also known as Euler's Factorization.

Previously, the result arrived at was a sum of four squares which can be expressed as the product of two sums of two squares, as shown below:

$$N = (x_1y_1)^2 + (x_1y_2)^2 + (x_2y_1)^2 + (x_2y_2)^2 = \left(x_1^2 + x_2^2\right)\left(y_1^2 + y_2^2\right)$$

Euler's factorization method [23] can be mathematically formulated as shown below:

$$N = (x_1y_1 - x_2y_2)^2 + (x_1y_2 + x_2y_1)^2 = (x_1y_1 + x_2y_2)^2 + (x_1y_2 - x_2y_1)^2$$

The parity (E: even; O: odd) of the squares can be used to find a factor, as presented in our earlier work [15].

$$P = (E_1 - E_2)/(O_1 - O_2)\, Q = (E_1 + E_2)/(O_1 - O_2)$$

We observe that the limitation of the sum of four squares is that it only applies to $P, Q \equiv 1 mod 4$, which relates to Fermat's Christmas Theorem discovered as early as 1640. Fermat's Christmas Theorem on the sum of two squares states that an odd prime can be expressed as $P = x^2 + y^2$ if $P \equiv 1 \ mod \ 4$.

The method outlined above is restricted to provide factors of semi-primes whose construction is of the form $N = PQ: \ 1 mod 4 = 1 mod 4 * 1 mod 4$ only. The solutions are limited in this case. It is observed that with $3 mod 4 = 1 mod 4 * 3 mod 4$, $3 mod 4 = 3 mod 4 * 1 mod 4$, $1 mod 4 = 3 mod 4 * 3 mod 4$, the solvable semi-primes cannot be resolved.

We now propose an extension to the Brahmagupta–Fibonacci identity outlined above by **not** considering the **sums** of squares for the two prime number construction of the semi-prime. Instead, we will now consider the difference of squares. Using the difference of squares allows all of the primes, both $1 mod 4$ and $3 mod 4$, to be expressed in this way. By considering the difference of squares all primes can be considered, and, as a consequence, all semi-prime constructions may now be considered (not just the $1 mod 4 = 1 mod 4 * 1 mod 4$ constructions).

The limitation of the sums of squares method of factorization is the key motivation in our work to consider the difference of squares. We then investigated with the following question in mind:

Can we modify the Brahmagupta–Fibonacci identity [33] so that it can be applied to the difference of squares?

If we can find such a method, then we posit that Fermat's factorization can be used.

$$N = x_1^2 - x_2^2 = (x_1 - x_2)(x_1 + x_2)$$

This removes the $1 mod 4$ constraint and includes all primes, namely $1 mod 4 and 3 mod 4$.

The modified Brahmagupta–Fibonacci identity can be applied to the product of two differences of two squares as shown below:

$N = PQ$, where P and Q are both odd primes.

$P = \left(\frac{P+1}{2}\right)^2 - \left(\frac{P-1}{2}\right)^2, Q = \left(\frac{Q+1}{2}\right)^2 - \left(\frac{Q-1}{2}\right)^2.$

$N = \left(\frac{N+1}{2}\right)^2 - \left(\frac{N-1}{2}\right)^2 trivial$

Further, this can be extended such that $N = PQ$, where P and Q are both odd primes.

$$N = \left(\left(\frac{P+1}{2}\right)\left(\frac{Q+1}{2}\right)\right)^2 - \left(\left(\frac{P+1}{2}\right)\left(\frac{Q-1}{2}\right)\right)^2 - \left(\left(\frac{P-1}{2}\right)\left(\frac{Q+1}{2}\right)\right)^2 + \left(\left(\frac{P-1}{2}\right)\left(\frac{Q-1}{2}\right)\right)^2$$

$$N = \left(\left(\frac{P+1}{2}\right)\left(\frac{Q+1}{2}\right) - \left(\frac{P-1}{2}\right)\left(\frac{Q-1}{2}\right)\right)^2 - \left(\left(\frac{P+1}{2}\right)\left(\frac{Q-1}{2}\right) - \left(\frac{P-1}{2}\right)\left(\frac{Q+1}{2}\right)\right)^2$$

$kN = a^2 - b^2 - c^2 + d^2, ad - bc = 0,$

$kN = (a-d)^2 - (b-c)^2.$

In Section 4, we showed that the convergence provides a value for k that can be bound within a search field of composites with reduced complexity to about 1/3 of the number that is considered for factorization. We acknowledge that this is now in a composite field rather than a search in a prime field. The attributes of k are now further constrained and the search in the composites now requires that additional constraints be met, as shown below:

$kN = a^2 - b^2 - c^2 + d^2$, $ad - bc = 0$,

$kN = (a - d)^2 - (b - c)^2$.

The above theoretical ideas pave the way for future research into extending the proposed algorithm with the aim of further reducing the search space in order to find a faster solution to the semi-prime factorization problem. However, comparative investigations on the difference of squares method are considered to be future work as they are beyond the scope of this paper.

7. Measures of Performance and Future Research

In this section, we identify suitable measures of performance for our proposed method and further improvement ideas in our ongoing research work. We consider the relationships between solutions of the form $kN = x^2 - y^2 + z^2,\ z :\ z = 0$ and how the solution is found in a reduced search space. However, many intermediate solutions exist and we look for relationships between the following:

$$k_1N = x_1^2 - y_1^2 + z_1^2, k_2N = x_2^2 - y_2^2 + z_2^2 \ldots k_nN = x_n^2 - y_n^2 + z_n^2$$

Further, we look for forms which fit into the modified Brahmagupta–Fibonacci identity.

7.1. Measures of Performance

Our ongoing research is motivated by comparing measures of performance for our method with some popular existing works. We summarize them below:

Lehman's method [31] looked for multipliers $k : 0 < k \leq \sqrt[3]{N} + 1$. This obtained a factoring time of $O\left(\sqrt[3]{N}\right)$.

The method presented here places a constraint on the lower bound of k by removing continued fraction convergences which are unlikely to find a suitable k value. We observe that a convergence appears approximately within one-third of the convergence and provides the upper and lower bound interval. A faster solution can be achieved by segmenting the interval to search in multi-processor platforms. This would be part of our ongoing research work considering other research ideas such as the difference of squares presented earlier.

It would be worthwhile to consider intermediate solutions of the form $kN = ad - bc = 0$, which may otherwise have been discarded, that provide solutions to $kN = a^2 - b^2 - c^2 + d^2$. Such criteria would restrict the number of candidates for k which are subjected to the limits of $\sqrt{kN}$.

Our experiments were conducted with a computer with the following specifications: Intel Xeon E5-2697 v2 2.7 GHz, 32 GB DDR3 memory. From our previous example (21,565,941,721,999,797,939,843,713,963, which is a 95-bit prime number) considered by Crandall and Pomerance [48] on pg307, we present the performance of our algorithm. A brute force search finds that 17,887/6172 leads to a factorization, with an exhaustive search time of 10 min 38 s.

Observe that $\frac{17887}{6172} \Rightarrow \frac{[31 \times 577]}{[2 \times 2 \times 1543]}, k = 4pq = 4 \times 17887 \times 6172 = 441594256$.

From the earlier example, the continued fraction length was 26 and the 11th convergence provided a suitable k value. The resulting factorization is shown below:

The continued fraction for $\frac{p}{q}$ is given as

1,000,000,000,000,037/21,565,941,721,999 => 46 2 1 2 2 2 2 1 1 1 3 8 13 1 263 1 5 38 1 4 1 2 8 2 1 1 2

Observe that the $11th$ *convergence of* $\left[\frac{p}{q}\right] = \frac{71548}{1543} = \frac{[2 \times 2 \times 31 \times 577]}{[1543]}$.

$$k = 4pq = 4 \times 71548 \times 1543 = 441594256$$

Recalling the brute force method also obtains

$$k = 4pq = 4 \times 17887 \times 6172 = 441594256$$

$$[k,\ x,\ y,\ z] = [441594256,\ 3085999998325641543,\ 1674472639,\ 0]$$

$$P1 = 1000000000000037,\ \ P2 = 21565941721999$$

A brute force search found the solution using the fraction 17,887/6172, whose factorizations are [31 577]/[2 2 1543], confirming the solution using our proposed method above.

7.2. Improvements and Future Research

Hart used $Y^2 = (\lceil\sqrt{4kN}\rceil)^2\ mod\ N$ as $4pqN$ which is already known [21]. We improved this by using $Y^2 = \left(\lceil\sqrt{4pqN}\rceil\right)^2 - 4pqN$. This helps in reducing the search space to arrive at a faster convergence of a solution to the semi-prime factorization problem.

Wiener's cryptographic attack against RSA using the continued fraction method [55] motivates many researchers to continue exploring faster approaches to factorization of RSA. Public key encryption uses Euler's totient function [56,57], which provides the basis for factoring algorithms that have a major impact on cybersecurity. Hence, past works on number theory, primality testing, and the implementation of algorithms [58–63] have been the backbone of this research work to arrive at a new method. Further, the speeding up of existing methods using more efficient methods and the adoption of more computing power, including parallelism [64–67], have fueled this ongoing research. Our research investigations have taken a modest step forward in this work and will continue to further reduce the search space for arriving at a solution to the semi-prime factorization problem more quickly. We hope that our research work on Hart's one line factorization method will stimulate interest in this method. In future work, we intend to investigate the use of our narrower definition of the parameterization of k to demonstrate further improvements in the Wiener attack. The scope of this investigation is well outside of this paper due to the limitations stated before.

8. Conclusions

This paper proposed a new approach of applying continued fractions to the one line factoring algorithm. Lower ordered convergences as representations of $\frac{P}{Q}$ that form continued fractions were applied to Hart's one line factoring algorithm. We illustrated our method using a worked example as well as by finding the factors of RSA250 successfully. The key contributions of this work that paves the way for further research are as follows:

1. We established that continued fraction convergences $\frac{p}{q}$ as representations of $\frac{P}{Q}$ provide solutions to Hart's one line factorization.
2. We explored relationships of the form $kN = x^2 - y^2 + z^2$ for convergence.
3. We investigated the suitability of the modified Brahmagupta–Fibonacci identity for factorization considering

$$kN = a^2 - b^2 - c^2 + d^2,\ ad - bc = 0 \qquad kN = (a-d)^2 - (b-c)^2$$

4. Our experimental findings include the RSA modulus, the ratio of $\frac{P}{Q}$ to be constrained by a particular number of bits.
   $P, Qbits = \frac{N}{2} bits \Rightarrow 1 < \frac{P}{Q} < 2\ 1 < \frac{p}{q} < 2$ such that $x^2 - y^2 = 4kN, k = pq$.
   k is still an area of interest and may yet yield deeper connections.

In this paper, we studied many correct solutions which formed the convergences of PQ that had not been explored previously. The results shown in this paper are quite important as we explored the relationship between the convergence of PQ and its relationship to k, which is used in Hart's one line factorization algorithm. Our significant contribution is that after about one third of the way through the convergences, solutions for k were found and all convergences after that resulted in valid solutions for k. Further, there is a scarcity of research exploring the attributes of k, and this paper provides a better understanding of its attributes.

Further, our practical implementation along with our findings motivated us to extend the study using the difference of squares. The theory on the difference of squares provides the relationship to the proposed approach and future extensions to our algorithm, motivating other researchers to consider such extensions to the existing work. We believe the observations and limitations of this will open future investigations as forms of ongoing research.

In Section 4, we showed that the convergence that provides a value for k can be bound to a search field of composites with reduced complexity to about $1/3$ of the number attempting to be factorized whilst acknowledging that this is now a search in a composite field rather than a search in a prime field. In Section 6, the attributes of k were further defined, thereby constraining the suitable composite candidates for k.

Our findings are unique in that only one other paper has been published regarding Hart's one line factoring algorithm since its publication in 2012.

It is hoped that the wider research community might find our results interesting and that further investigations into Hart's work might be encouraged.

Author Contributions: Conceptualization, A.O.; methodology, A.O. and S.V.; formal analysis, A.O.; validation, S.V.; investigation, A.O. and S.V.; resources, A.O.; data curation, A.O.; writing—original draft preparation, A.O.; writing—review and editing, S.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Rivest, R.L.; Shamir, A.; Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. [CrossRef]
2. Menezes, A.J.; van Oorschot, P.C.; Vanstone, S.A. *Handbook of Applied Cryptography*; CRC Press: New York, NY, USA, 1997.
3. Rabah, K. Secure implementation of message digest, authentication and digital signature. *Inform. Technol. J.* **2005**, *4*, 204–221. [CrossRef]
4. Rabin, M.O. *Digital Signature and Public-Key Functions as Intractable as Factorization*; MIT Laboratory of Computer Science: Cambridge, MA, USA, 1979.
5. Leutwyler, K. Superhack: Forty quadrillion years early, a 129-digit code is broken. *Sci. Am.* **1994**, *271*, 17–20.
6. Cowie, J.; Dodson, B.; Elkenbracht-Huizing, R.M.; Lenstra, A.K.; Montgomery, P.L.; Zayer, J. A World Wide Number Field Sieve Factoring Record: On to 512 Bits. In *Advances in Cryptology-Asiacrypt '96*; Kim, K., Matsumoto, T., Eds.; Springer: Berlin/Heidelberg, Germany, 1996; pp. 382–394.
7. Lehmer, D.H.; Powers, R.E. On Factoring Large Numbers. *Bull. Am. Math. Soc.* **1931**, *37*, 770–776. [CrossRef]
8. Morrison, M.A.; Brillhart, J. A method of factorization and the factorization of F7. *Maths. Comp.* **1975**, *29*, 183–205.
9. Pomerance, C. Analysis and Comparison of Some Integer Factoring Algorithms. In *Computational Methods in Number Theory*; Lenstra, H.W., Jr., Tijdeman, R., Eds.; Mathematisch Centrum: Amsterdam, The Netherlands, 1982; pp. 89–139.
10. Lenstra, A.K.; Manasse, M.S. Factoring with two large primes. *Maths. Comp.* **1994**, *63*, 785–798. [CrossRef]
11. Pollard, J.M. Factoring with cubic integers. *Lect. Notes Math.* **1993**, *1554*, 4–10.
12. Shor, P.W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.* **1997**, *26*, 1484–1509. [CrossRef]
13. Ambedkar, B.R.; Bedi, S.S. A New Factorization Method to Factorize RSA Public Key Encryption. *Int. J. Comput. Sci. Issues* **2011**, *8*, 242–247.

14. Wu, L.; Cai, H.J.; Gong, Z. The Integer Factorization Algorithm with Pisano Period. *IEEE Access* **2019**, *7*, 167250–167259. [CrossRef]
15. Overmars, A.; Venkatraman, S. A Fast Factorisation of Semi-Primes Using Sum of Squares. *Math. Comput. Appl.* **2019**, *24*, 62. [CrossRef]
16. Rutkowski, E.; Houghten, S. Cryptanalysis of RSA: Integer Prime Factorization Using Genetic Algorithms. In Proceedings of the 2020 IEEE Congress on Evolutionary Computation (CEC), Glasgow, UK, 19–24 July 2020; pp. 1–8.
17. Overmars, A.; Venkatraman, S. New Semi-Prime Factorization and Application in Large RSA Key Attacks. *J. Cybersecur. Priv.* **2021**, *1*, 660–674. [CrossRef]
18. Lenstra, H.W., Jr. Factoring integers with elliptic curves. *Ann. Math.* **1987**, *126*, 649–673. [CrossRef]
19. Pomerance, C. The quadratic sieve factoring algorithm. In Proceedings of the EUROCRYPT 84 Workshop on Advances in Cryptology: Theory and Application of Cryptographic Techniques, (WACTACT'85), Paris, France, 9–11 April 1984; Springer: New York, NY, USA, 1985; pp. 169–182.
20. Pomerance, C. A tale of two sieves. *Not. Aim* **1996**, *43*, 1473–1485.
21. Lenstra, A.K.; Lenstra, H.W., Jr. (Eds.) *The Development of the Number Field Sieve*; LNM; Springer: Berlin/Heidelberg, Germany, 1993; Volume 1554.
22. Adrian, D.; Bhargavan, K.; Durumeric, Z.; Gaudry, P.; Green, M.; Halderman, J.A.; Heninger, N.; Springall, D.; Thomé, E.; Valenta, L.; et al. Imperfect forward secrecy: How Diffie-Hellman fails in practice. *Commun. ACM* **2018**, *62*, 106–114. [CrossRef]
23. Kleinjung, T.; Aoki, K.; Franke, J.; Lenstra, A.K.; Thomé, E.; Bos, J.W.; Gaudry, P.; Kruppa, A.; Montgomery, P.L.; Osvik, D.A.; et al. Factorization of a 768-bit RSA modulus. In *CRYPTO 2010. LNCS.*; Rabin, T., Ed.; Springer: Berlin/Heidelberg, Germany, 2010; Volume 6223, pp. 333–350. [CrossRef]
24. Boudot, F.; Gaudry, P.; Guillevic, A.; Heninger, N.; Thomé, E.; Zimmermann, P. Comparing the difficulty of factorization and discrete logarithm: A 240-digit experiment. In *Advances in Cryptology-CRYPTO 2020*; Micciancio, D., Ristenpart, T., Eds.; Springer: Berlin/Heidelberg, Germany, 2020; pp. 62–91.
25. Overmars, A.; Ntogramatzidis, L. A new parameterisation of Pythagorean triples in terms of odd and even series. *arXiv* **2015**, arXiv:1504.03163.
26. Overmars, A.; Ntogramatzidis, L.; Venkatraman, S. A new approach to generate all Pythagorean triples. *AIMS Math.* **2019**, *4*, 242–253. [CrossRef]
27. Venkatraman, S.; Overmars, A. New method of prime factorisation based attacks on RSA Authentication in IoT. *Cryptography* **2019**, *3*, 20. [CrossRef]
28. Overmars, A.; Venkatraman, S. Mathematical Attack of RSA by Extending the Sum of Squares of Primes to Factorize a Semi-Prime. *Math. Comput. Appl.* **2020**, *25*, 63. [CrossRef]
29. Overmars, A.; Venkatraman, S. A New Method for Factorizing Semi-primes Using Simple Polynomials. In Proceedings of the 3rd International Conference on Research in Applied Science, Munich, Germany, 6–8 November 2020.
30. Venkatraman, S.; Overmars, A. IoT Authentication and Security Challenges. In Proceedings of the 3rd International Conference on Research in Applied Science, Munich, Germany, 6–8 November 2020; Diamond Scientific Publishing: Vilnius, Lithuania, 2020.
31. Lehman, R.S. Factoring Large Integers. *Math. Comput.* **1974**, *28*, 637–646. [CrossRef]
32. Hart, W.B. A one line factoring algorithm. *J. Aust. Math. Soc.* **2012**, *92*, 61–69. [CrossRef]
33. Fibonacci, L. *Liber Quadratorum (The Book of Squares)·Liber Minoris Guise (n.d.)*; Sigler, L.E., Translator; Academic Press: Orlando, FL, USA; Cambridge, MA, USA, 1987; ISBN 978-0-12-643130-8.
34. Brent, R.P.; Pollard, J.M. Factorization of the eighth Fermat number. *Maths. Comput.* **1981**, *36*, 627–630. [CrossRef]
35. Mahoney, M.S. *The Mathematical Career of Pierre de Fermat*, 2nd ed.; Princeton University Press: Princeton, NJ, USA, 1994; pp. 1601–1665.
36. Northshield, S. A Short Proof of Fermat's Two-square Theorem. *Am. Math. Mon.* **2020**, *127*, 638. [CrossRef]
37. McKee, J.F. Speeding Fermat's factoring method. *Math. Comput.* **1999**, *68*, 1729–1737. [CrossRef]
38. Brent, R.P. Factorization of the tenth Fermat number. *Maths. Comp.* **1999**, *68*, 429–451. [CrossRef]
39. Wu, M.E.; Tso, R.; Sun, H.M. On the improvement of Fermat factorization using a continued fraction technique. *Future Gener. Comput. Syst.* **2014**, *30*, 162–168. [CrossRef]
40. Somsuk, K.; Tientanopajai, K. An improvement of fermat's factorization by considering the last m digits of modulus to decrease computation time. *Int. J. Netw. Secur.* **2017**, *19*, 99–111.
41. Kraitchik, M. *Recherches sur la Theorie des Nombres*; Gauthier Villar: Paris, France, 1929; Volume II.
42. Knuth, D.E. *The Art of Computer Programming*, 2nd ed.; Addison-Wesley: Reading, MA, USA, 1981; Volume 2, p. 383.
43. Pomerance, C.; Wagstaff, S.S., Jr. Implementation of the continued fraction integer factoring algorithm. *Congr. Numer.* **1983**, *37*, 99–118.
44. Pomerance, C.; Smith, J.W.; Wagstaff, S.S., Jr. New ideas for factoring large integers. In *Advances in Cryptology*; Springer: Boston, MA, USA, 1984; pp. 81–85.
45. Williams, H.C.; Shallit, J.O. Factoring integers before computers. *Proc. Symp. Appl. Math.* **1994**, *48*, 481–531.
46. Shanks, D. SQUFOF Notes. Manuscript, 27 Pages. 1982. Available online: https://homes.cerias.purdue.edu/~ssw/shanks.pdf (accessed on 24 January 2023).
47. Silverman, R.D. The multiple polynomial quadratic sieve. *Math. Comp.* **1987**, *48*, 329–340. [CrossRef]
48. Crandall, R.E.; Pomerance, C. *Prime Numbers. A Computational Perspective*; Springer: New York, NY, USA, 2001.

49. Sierpinski, W. *Elementary Theory of Numbers*; Polish Scientific Publishers: Warszawa, Poland, 1964; pp. 282–314.
50. Budiman, M.A.; Rachmawati, D. Using random search and brute force algorithm in factoring the RSA modulus. *Data Sci. J. Comput. Appl. Inform.* **2018**, *2*, 45–52. [CrossRef]
51. Nemec, M.; Sys, M.; Svenda, P.; Klinec, D.; Matyas, V. The Return of Coppersmith's Attack: Practical Factorization of Widely Used RSA Moduli. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. CCS'17, Dallas, TX, USA, 30 October–3 November 2017. [CrossRef]
52. Eyal, I.; Sirer, E. Majority is not enough: Bitcoin mining is vulnerable. In *Financial Cryptography and Data Security. FC 2014. Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2014; Volume 8437. [CrossRef]
53. Tiwari, A. Chapter 14—Cryptography in blockchain. In *Distributed Computing to Blockchain*; Pandey, R., Goundar, S., Fatima, S., Eds.; Academic Press: Cambridge, MA, USA, 2023; pp. 251–265. ISBN 978032396146.
54. Ikeda, K. Chapter Seven—Security and Privacy of Blockchain and Quantum Computation. In *Advances in Computers*; Raj, P., Deka, G.C., Eds.; Elsevier: Amsterdam, The Netherlands, 2018; Volume 111, pp. 199–228.
55. Wiener, M. Cryptanalysis of short RSA secret exponents. *IEEE Trans. Inf. Theory* **1990**, *160*, 553–558. [CrossRef]
56. Erdós, P. On the Normal Number of Prime Factors of P-1 and Some Related Problems Concerning Euler's Ø-Function. *Q. J. Math.* **1935**, *6*, 205–213. [CrossRef]
57. McKee, J.F. Turning Euler's Factoring Method into a Factoring Algorithm. *Bull. Lond. Math. Soc.* **1996**, *28*, 351–355. [CrossRef]
58. Pollard, J.M. Theorems on factorization and primality testing. *Proc. Cambridge Phil. Soc.* **1974**, *76*, 521–528. [CrossRef]
59. Lenstra, A.K.; Lenstra, H.W., Jr. Algorithms in number theory. In *Handbook of Theoretical Computer Science: Algorithms and Complexity*; Van Leeuwen, J., Ed.; Elsevier: Amsterdam, The Netherlands, 1990; Volume A, pp. 673–715.
60. Montgomery, P.L. A survey of modern integer factorization algorithm. *Maths. Comput.* **1994**, *7*, 337–338.
61. Bahig, H.M.; Nassr, D.I.; Mahdi, M.A.; Bahig, H.M. Small Private Exponent Attacks on RSA Using Continued Fractions and Multicore Systems. *Symmetry* **2022**, *14*, 1897. [CrossRef]
62. Nitaj, A.; Kamel Ariffin, M.R.; Hanisah Adenan, N.N.; Chien Lau, T.S.; Chen, J. Security Issues of Novel RSA Variant. *IEEE Access* **2022**, *10*, 53788–53796. [CrossRef]
63. Bansimba, G.R.; Babindamana, R.F.; Bossoto, B.G.R. A Continued Fraction-Hyperbola based Attack on RSA cryptosystem. *arXiv* **2023**, arXiv:2304.03957.
64. Bahig, H.M. Speeding Up Fermat's Factoring Method using Precomputation. *Ann. Emerg. Technol. Comput.* **2022**, *6*, 51–60. [CrossRef]
65. Montgomery, P.L. Speeding the pollard and elliptic curve methods of factorization. *Maths. Comput.* **1987**, *48*, 243–264. [CrossRef]
66. Dixon, B.; Lenstra, A.K. Massively parallel elliptic curve factoring. In Proceedings of the Eurocrypt '92, Lecture Notes in Computer Science, Balatonfüred, Hungary, 24–28 May 1992; Springer: Berlin/Heidelberg, Germany, 1993; pp. 183–193.
67. Eldershaw, C.; Brent, R.P. Factorization of large integers on some vector and parallel computers. *Proc. Neural Parallel Sci. Comput.* **1995**, *1*, 143–148.