DOI 10.29042/2018-2996-3003

Helix Vol. 8(1): 2996-3003

Analysis of the Lenstra Elliptic Curves Factorization Method

[*1] Albert I. Galiev, [2] Shamil T. Ishmukhametov, [3] Ramilya G. Rubtsova

[1, 2, 3] Kazan Federal University

Email: *[*1]crowbar_al@mail.ru, Shamil.Ishmukahametov@kpfu.ru, Ramilya.Rubtsova@kpfu.ru*

Received: 15th December 2017, Accepted: 20th December 2017, Published: 31st December 2017

Abstract

The problem of factoring a composite natural number into the product of prime factors is a hard computational problem. This problem lies in the base of the well-known open-key RSA cyphering algorithm. In the paper we investigate Lenstra's Elliptic Curves Factoring Algorithm, which is the third (after the Number Field Sieve and the Quadratic Sieve) by speed algorithm in the world classification of factoring algorithms. Moreover, the speed of this algorithm depends mostly on the size of the minor factor, so it can be applied to very large record numbers. Original Lenstra's algorithm of 1987 year consisted of one stage while later it was shown that in many cases it is more effectively to use the two-stage version.But when we are able to use multi-processors systems, we can arrange parallel computations which use different versions of the basic algorithm and are more effective. In the paper, we consider several improvements tothe LenstraBasic Algorithm and compare them by speed. In particular, we study the special Montgomery presentation of elliptic curves and show that they give a considerable acceleration of the factoring procedure. We measure the speed of four realizations of Lenstra's algorithm to choose the best one.

Keywords: Factoring Problem, RSA Algorithm, Elliptic Curves, Lenstra's Elliptic Curves Method, Montgomery Presentation of Elliptic Curves.

Introduction

Elliptic curves have become very popular in number theory and algebraic geometry in connection with their applications to cryptography. Such applications became possible after independent publications by Neil Koblits[1] and Victor Miller [2]in 1985.These works made the beginning of a series of studies on the use of elliptical curves in cryptography

A detailed exposition of the theory of elliptic curves and their applications in cryptography can be found in the article [3] and the monograph [4]. The monograph [5] describes cryptographic protocols using elliptic curves. The reason why elliptic curves have received interest from cryptographers is that the problem of discrete logarithm on elliptic curves considered over finite fields (that is, computing the multiple x for two given points of the curve P, Q, $Q = xP$), is much more difficult to solve from the computational viewpoint than the analogous problem for the factorization of integers or the calculation of the discrete logarithm in finite fields. This led to the fact that with less computational costs and greater cryptographic stability, it was possible to accomplish the tasks of encryption and constructing digital signatures, using elliptic curves.

Hendrik Lenstra in his article of 1987 [6] gave a description of a new subexponential algorithm of factorization, which was included in the three fastest algorithms of factorization. The advantage of this algorithm is also that its computational complexity depends only on the size of the minimum divisor, and not on the size of the factorized number.

S. T.Ishmukhametovin the monograph [7] gives a detailed description of the main algorithms of factorization, known to date.

This article considers four modifications of the Lenstra algorithm: the classical algorithm in affine coordinates, proposed by Lenstra and consisting of one stage, the algorithm with two stages, the algorithm in projective coordinates, and the Montgomery method. The study of this article continues the study of the article [8].

Elliptic Curves

The general equation of an elliptic curve has the form

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6, \quad (1)$$

Where the variables x, y and constants$a_1, \dots, a_6$are considered in a finite or in finite field. Elliptic curves in cryptography are considered only over finite fields of $\mathbf{F_p}$ for some prime number p, or the fields$\mathbf{F}_q$, where $q = p^k$for $k \geq 2$.. The number $p$is called*the characteristic* of the field.

If the characteristic of the field is greater than 3, then equation (1) can be reduced to the form

$$y^2 = x^3 + ax + b, \quad (1)$$

which is called the Weierstrass equation.

In the future we will use this form. Define the set of points of an elliptic curve$E(F_q)$:

$$E(F_q) = \{\infty\} \cup \{(x, y) \in F_q \times F_q | y^2 = x^3 + ax + b \ (mod q)\}.$$

By definition, this set always contains a singular point $\{\infty\}$, which is called a point at infinity.

One of the most important parameters of the elliptic curve is the discriminant:

$$\Delta = -16(4a^3 + 27b^2).$$

In order that the elliptic curve not to have singular points, the condition$\Delta \neq 0$ must be satisfied. In what follows, we shall consider only such elliptic curves.

Group Laws

Now consider the operations that can be performed on the points of the elliptic curve. Consider two points$P_1 = (x_1, y_1)$и$P_2 = (x_2, y_2)$, belonging to the curve E, given by the equation (2). Define a new point P_3 such, that

$$P_1 + P_2 = P_3$$

Using the geometric interpretation:

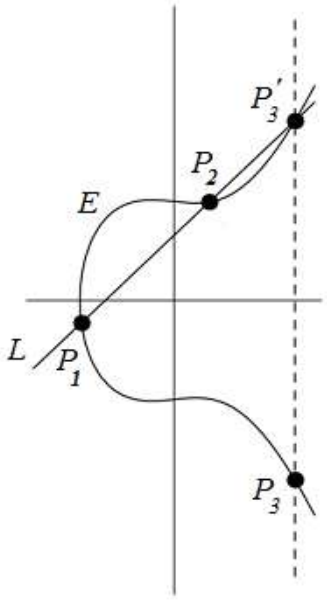

Figure 1. Addition of Two Points of EC in Geometric Interpretation.

The basic operations with the points of the EC:

Addition of two points.

If the pointsP_1, P_2differ from∞, $P_1 \neq P_2$and $x_1 \neq x_2$, that is, the straight line$L$: $y = kx + d$isnot vertical, the coefficients $k$and$d$can be represented in the form:

$$k = \frac{y_2 - y_1}{x_2 - x_1},$$

$$d = y_1 - kx_2.$$

Substituting (4) and (5) in the equation of the straight line L, and then the right side of the equation into the equation of the curve (3), one can obtain the coordinates of the point P_3:

$$\begin{cases} x_3 = k^2 - x_1 - x_2 \\ y_3 = k(x_1 - x_3) - y_1 \end{cases}.$$

If$x_1 = x_2$and $y_1 \neq y_2$, the straight line$L$is vertical. Thus, $P_1 + P_2 = \infty$.

Thepoint{∞}is a neutral element of A belian group E for addition, that is, for any point $P \in E$:

$$P + \infty = P.$$

An inverse element for the point$P = (x, y)$is the point$-P = (x, -y)$:

$$P + (-P) = \infty.$$

2) Duplication of the point.

If$P_1 = P_2 = (x_1, y_1)$,theline$L$isthetangentto$E$inthepointP_1. Differentialequations(2) at$x$allows to determinean angle of inclination of the straight line L:

$$2y\frac{dy}{dx} = 3x^2 + a, \quad \text{hence} \quad k = \frac{3x_1^2 + a}{2y_1}. \tag{7}$$

Then the point$P_3 = 2P_1$is determined by analogy:

$$\begin{cases} x_3 = k^2 - 2x_1 \\ y_3 = k(x_1 - x_3) - y_1 \end{cases}. \tag{8}$$

If$y_1 = 0$, then the straight line$L$is vertical, that is,$P_1 + P_2 = \infty$.

3) Multiplication of the point by the number $k \in N$.

Let the given point is$P = (x, y)$.It is necessary to find$Q = k \cdot P$.

Forthispurposeitisrequiredtoconvertdecimalnotation ofthe number $k$to binary notation, that is, to express$k$as $b_{n-1}, b_{n-2}, \ldots, b_0$, where$b_i \in \{0,1\}$for any$i = \overline{0, n-1}$.

Thus, the operation of multiplying by a number is expressed through successive operations of addition and doubling of points. At the step I of this algorithm, the point$2^i \cdot P$, obtained from the previous one by doubling, is calculated. Further, this point is added to the resultant if $b_i = 1$.

Elliptic Curves in Projective Coordinates

Since the operation of dividing by a number in a finite field is reduced to multiplying by the inverse element of a given number, it is necessary to use the extended Euclidean algorithm to find this inverse element. This operation is laborious, therefore it is suggested to use projective coordinates. In this case the point(x, y) given in affine coordinates in the finite field K corresponds to the equivalence class in projective coordinates:

$$(\lambda x, \lambda y, \lambda z), \lambda \in K.$$

There an equivalence class$(0,1,0)$corresponds to the point∞. In this case, the equation (3) is transformed into the form:

$$y^2z = x^3 + axz^2 + bz^3\ (modq). \tag{9}$$

When transferring from affine coordinates to projective ones, the following transformation is used:

$$(x, y) \to (x, y, 1).$$

In the reverse transition from projective to affine under the condition z ≠ 0, the following transformation is performed:

$$(x, y, z) \to (x/z, y/z).$$

Definetheoperationsoftwopoints: $(x_1, y_1, z_1)$and(x_2, y_2, z_2).

$$(x_1, y_1, z_1) + (x_2, y_2, z_2) = (x_3, y_3, z_3)$$

$$\begin{cases} T_0 = y_1z_2,\ T_1 = x_1z_2,\ T_2 = x_2z_1 \\ A = y_2z_1 - T_0,\ B = T_2 - T_1,\ A_2 = A^2,\ B_2 = B^2,\ B_3 = B^3 \\ T_3 = B_2T_2, E = z_1z_2, F = A_2E - B_3 - 2T_3 \\ x_3 = BF \\ y_3 = A(T_3 - F) - T_0B_3 \\ z_3 = B_3E \end{cases}$$

By complexity, this operation is evaluated as 12M+2S (12 multiplications and 2 squarings).

Similarly, the operation of doubling the point $(x_1, y_1, z_1)$is accomplished.

$$2(x_1, y_1, z_1) = (x_3, y_3, z_3)$$

$$\begin{cases} T_0 = x_1^2,\ T_1 = z_1^2,\ A = aT_1 + 3T_0 \\ B = 2y_1z_1,\ B_2 = B^2,\ E = y_1B,\quad E_2 = E^2 \\ F = (x_1 + E)^2 - T_0 - E_2,\ G = A^2 - 2F \\ x_3 = GB \\ y_3 = A(F - G) - 2E_2 \\ z_3 = BB_2 \end{cases}$$

Bycomplexity, thisoperationis estimated as 5M+6S.

Lenstra Factorization Algorithm

The Lenstra factorization algorithm, known as the Elliptic Curve Factorization Method (ECFM), is a modification of the well-known (p-1) Pollard method.In it, instead of the operation of raising a certain number to the power k modulo of the factorized number n, the operation of multiplying some random point of the elliptic curveP_0 by the number k in the finite set Z_n is performed. Since the set Z_n is not a field, it does not always define the inverse element for modulus division, which is necessary for calculating the sum of points.An elliptic curve with the equation$y^2 = x^3 + ax + b$on the set Z_n, where n- a composite number, will be called pseudocrit.

Let Nbe a factorizable number, and p - its least divisor. The essence of the algorithm is as follows:

The random numbers$a, b,$are chosenand the point P_0 on the pseu curve$EC(a, b, N)$is chosen.

Someboundary$B > 1$ischosen.

The starting point $P_0$is multiplied by all prime numbers and their degrees less than$B$. Recall that this operation consists of procedures of adding points or doubling them. Usually the operation of this step is performed using the Eratosthenessieve.

LetP_1be the final point. We calculate the greatest common divisor $GCD(n, u) = d$, where$u$is an arbitrary coordinate ofP_1, and if $d > 1$, then d is a nontrivial divisor of n.

If, as a result of calculating step 4, the value of GCD is 1, then the value of boundary$B$is too small. In this case, it is necessary to increase $B$and continue the calculation.

This is the scheme of the original Lenstra algorithm, consisting of a single stage.

We present the very algorithm, consisting of two stages. In the first stage, we do the same as in Algorithm 1. LetP_1be the final point obtained in the first stage, and GCD $d = 1$. In the second stage, the following isproceeded:

The constant $B_1 > B$is selected. Next, we find all prime numbers $B \le q_i < B_1$, $i = 0,1,2,...$

We compute the point $Q_0 = q_0P_1$.

The points$2P_1, 4P_1, ..., 2kP$are calculated for some small k.

We compute the points $Q_i = q_iP_1$, sequentially, performing the transition to the new point $Q_{i+1}$by one addition:

$$Q_{i+1} = Q_i + (q_{i+1} - q_i)P_1,$$

using one of the points calculated in step 3as a second point.

After each computation ofQ_{i+1}, we compute $d_{i+1} = d_i \cdot x_{i+1}\ mod\ N$, where x_{i+1} is the x-coordinate of the pointQ_{i+1} (any other coordinatecan be taken), $d_0 = 1$by definition.

After a given number of steps (for example, 100), we compute $GCD(N, d_i)$. If this GCDlies in between from 2 to N-1, then the divisor is found.

Estimation of the Efficiency of the Lenstra Factorization Algorithm

This algorithm is probabilistic, so the probability of its success depends to a large extent on the parameters B, B_1. In practice, the empirical dependence of these parameters on the length of the input number is used. For example, with a length N of 15 characters, $B = 2000$is usually taken, while for a length of 30 characters $B = 250000$ is used.

The value ofB_1 is chosen so that it must always be much greater than B.

If we evaluate the speed of the algorithm, then it should be mentioned that the ECM algorithm is subexponential, and, therefore, works faster than any exponential factorization algorithm. In this case, it is the third in speed after the method of quadratic sieve (QS) and the method of numerical field sieve (NFS). ECM is most effective for finding the dividers of 20 to 25 characters in length.

The expected complexity of the factorization algorithm on elliptic time curves is estimated as

$$O\left(\exp\sqrt{\left(2+o(1)\right)\ln p\ln(\ln \mathrm{p})}\right),$$

where$p$is the smallest of the divisors of the number n, o (1) is the term that tends to zero as $\mathrm{p}\to\infty$. In other words, this can be written in the form of an L-notation

$$L_p\left[1/2;\sqrt{2}\right].$$

Adding the second stage of the algorithm gives a significant increase in the speed of its work [8].

The Montgomery Method

Since it is necessary to resort to division moduloin calculating points in affine coordinates, instead of the equation (2) one can use a curve of the form

$$by^2 = x^3 + ax^2 + x,$$

which is called the Montgomery equation. This reduces the number of arithmetic operations. If we consider this equation in projective coordinates, then we can obtain

$$by^2z = x^3 + ax^2z + xz^2.(11)$$

Here one can neglect the coordinate y and consider the point as a class$(x,0,z)$. To generate the curve and the point$(x_1,0,z_1)$ (x_1,0, z_1), one can use the following formulas [2]:

$$u=\sigma^2-5, v=5\sigma,$$

$$x_1=u^3, z_1=v^3,$$

$$a=\frac{A_n}{A_d}-2=\frac{(v-u)^3(3u+v)}{4u^3v}-2,\ b=0,$$

where$\sigma\notin\{0,1,5\}$.

Now for doubling the point$P=(x_0,z_0)$it is necessary to compute

$$\begin{cases} x=4A_d(x_0+z_0)^2(x_0-z_0)^2 \\ z=(4A_d(x_0-z_0)^2+tA_n)t \end{cases}, \text{ где } t=(x_0+z_0)^2-(x_0-z_0)^2.$$

To add the points$P_1=(x_1,z_1)$and$P_2=(x_2,z_2)$it is necessary, first of all, to compute $P_0=P_1-P_2=(x_0,z_0)$, and then

$$\begin{cases} x=((x_1-z_1)(x_2+z_2)+(x_1+z_1)(x_2-z_2))^2z_0 \\ z=((x_1-z_1)(x_2+z_2)-(x_1+z_1)(x_2-z_2))^2x_0 \end{cases}.$$

A detailed exposition of the Montgomery algorithm can be found in [9].

Results and Discussion

To study the efficiency, factorization for different lengths of special input numbers obtained by the RSA methodwas carried out. All studies were carried out on a computer equipped with the AMD A6-6310 processor that ran at 2,2 GHz. Furtherthe results of these studies are recorded in Table 1. It should be noted that each number in this table is the average result obtained by performing n = 10 experiments. All units are expressed in milliseconds.

Table 1. The Speed of Algorithm Working with RSA-Numbers of Different Length, ms.

Input number length, bit (register length)	One-stage algorithm	Two-stage algorithm	Algorithm in projective coordinates	The Montgomery method
20(6)	3,3	2,7	3,2	3
30(9)	4,93	5,5	9,03	2,7
40(12)	56	57	48	10
50(15)	162	145	373	32
60(18)	1390	910	816	153
70(21)	5330	4649	4292	666
80(24)	12113	8655	7135	1413
90(27)	17134	14314	11907	3890
100(30)	46174	41703	35171	9594
110(33)	148156	134668	108324	21816
120(36)	592624	538672	425296	66417

Knowing these data, we can construct a graph of the dependence of the speed of the operation of the algorithms on the length of the input numbers. Below are shown these graphs for numbers up to 100 and 120 bits (Figures 2.1 and 2.2).

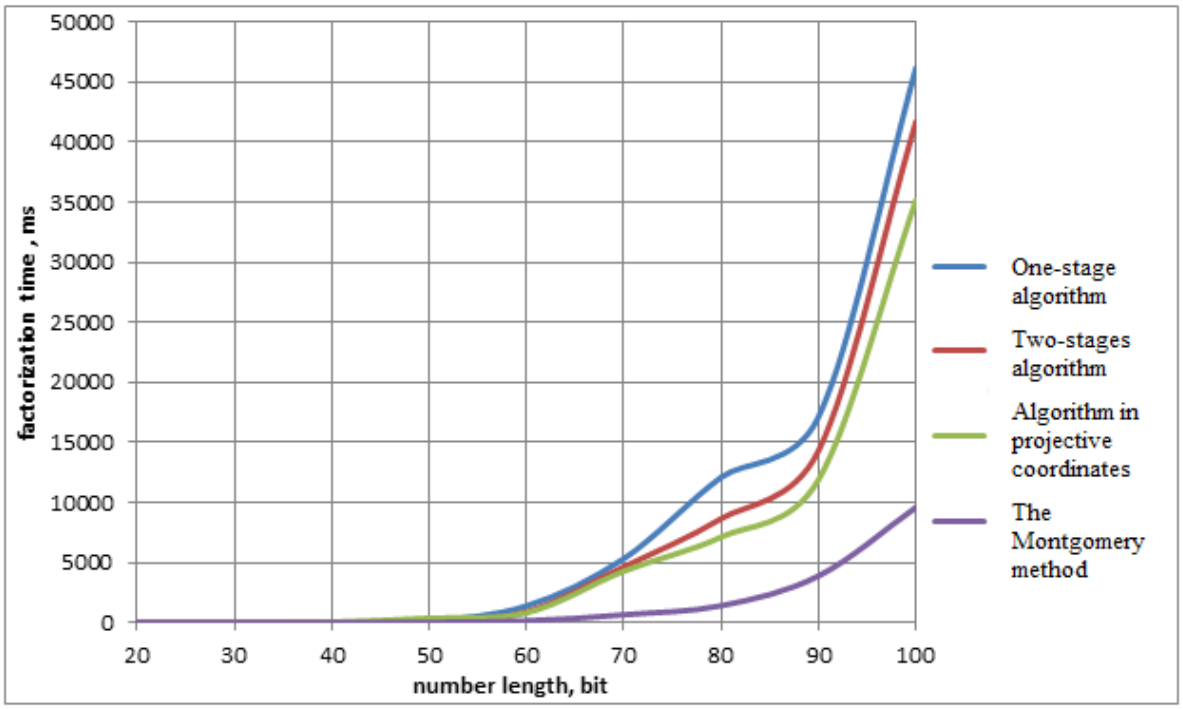

Figure 2.1. The Speed of Working of the Algorithms for the RSA-Numbers up to 100 bit.

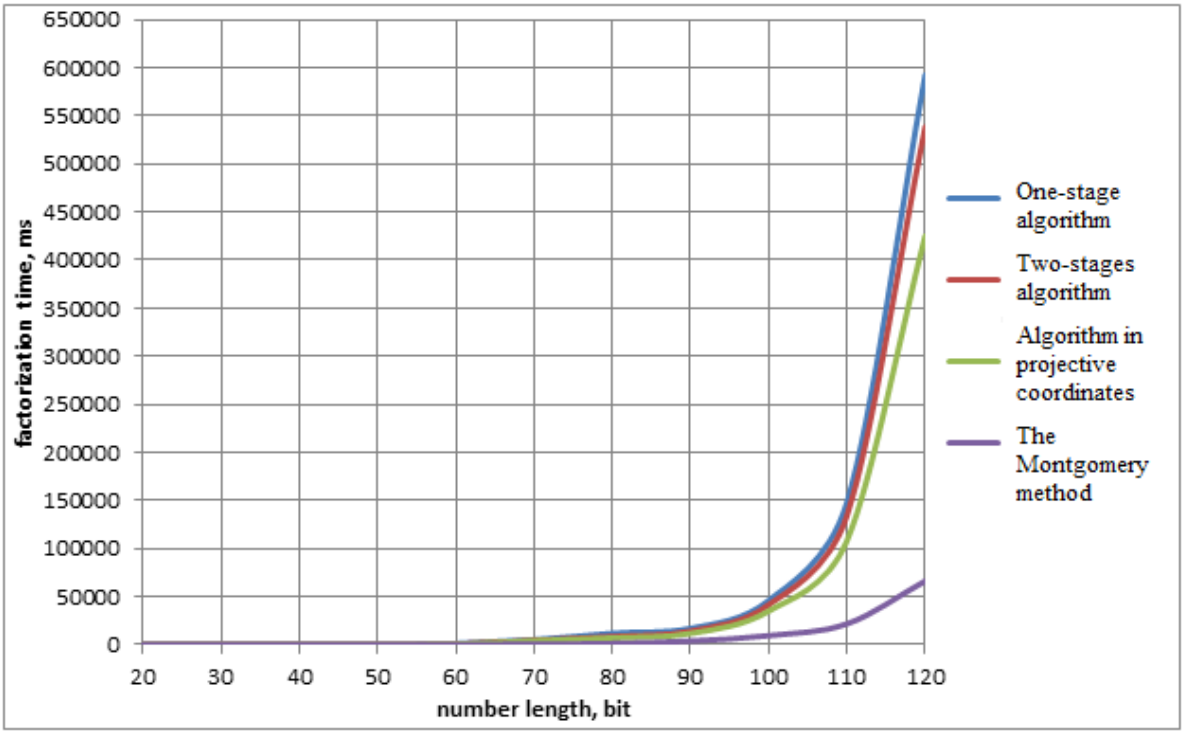

Figure 2.2. TheSpeedofWorkingof the Algorithms for the RSA-Numbersup to 120 bits.

From these graphs, we can say that the Montgomery method is many times faster than the classical method. The use of the second stage in the classical method gives a small increase in the speed of work, but not much. This can be explained by the fact that the probability of success of iteration in the

realization with two stages is higher than in the realization with a single stage, but iteration with two stages is performed longer under the same constraint parameters. Using conventional projective coordinates gives a slight advantage over the affine ones. Now consider the Montgomery method for the numbers up to 170 bits (51 characters). Data are represented in Table 2 and Figure 3.

Table 2. The Speed of Working of the Montgomery Method on the RSA-Numbers of Different Length.

Number, bit(orders)	20 (6)	30 (9)	40 (12)	50 (15)	60 (18)	70 (21)	80 (24)	90 (27)
Time, ms	3	2,7	10	32	153	666	1413	3890
Number, bit (orders)	100 (30)	110 (33)	120 (36)	130 (39)	140 (42)	150 (45)	160 (48)	170 (51)
Time, ms	9594	21816	66417	133530	474705	605630	1568786	9989155

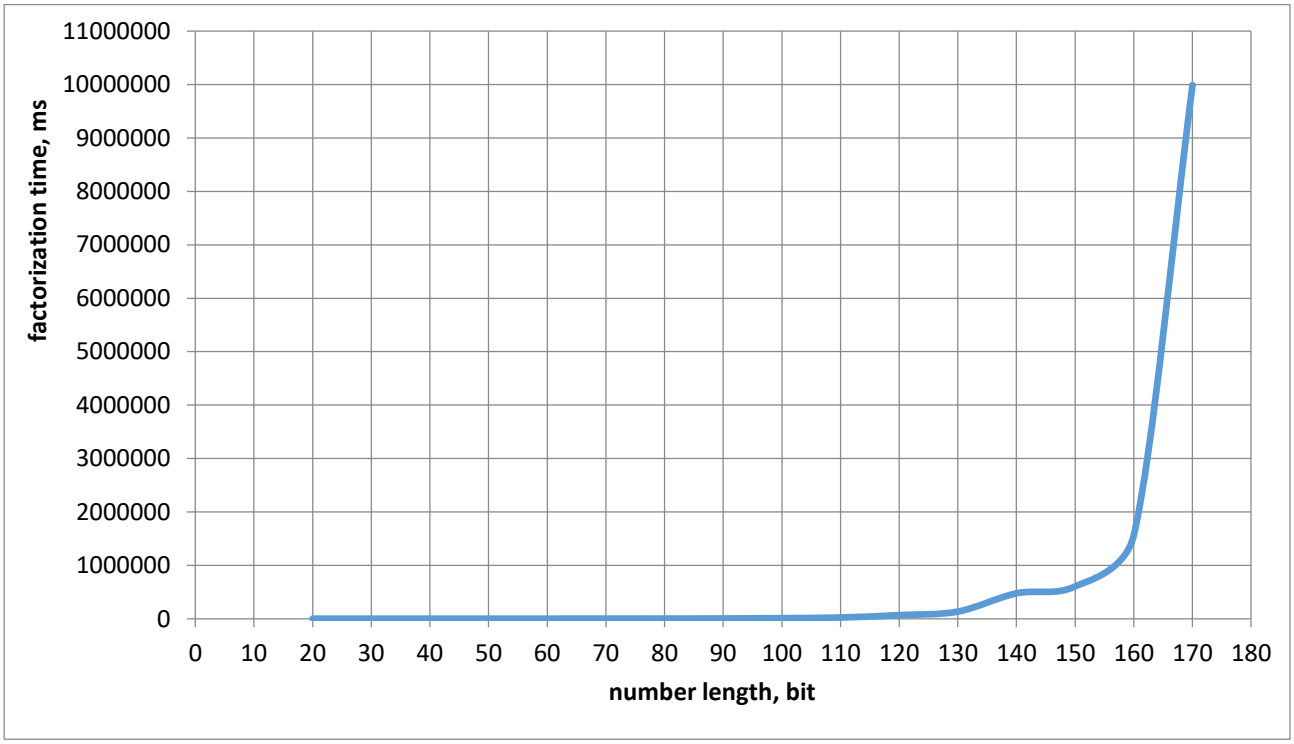

Figure 3. The Speed of Working of the Montgomery Method for the RSA-Numbers up to 170 bit.

Thus, using the Montgomery method, thenumber of 170-bit (51-bit) with the divisor of 85-bit (26 bits) was factorized for the average time $t = 9989155$ ms $\approx 2\ h$ 40 minutes.

Now consider the operation of the algorithm for the numbers of the same length, but having different lengths of the smallest divisor (Figure 4).

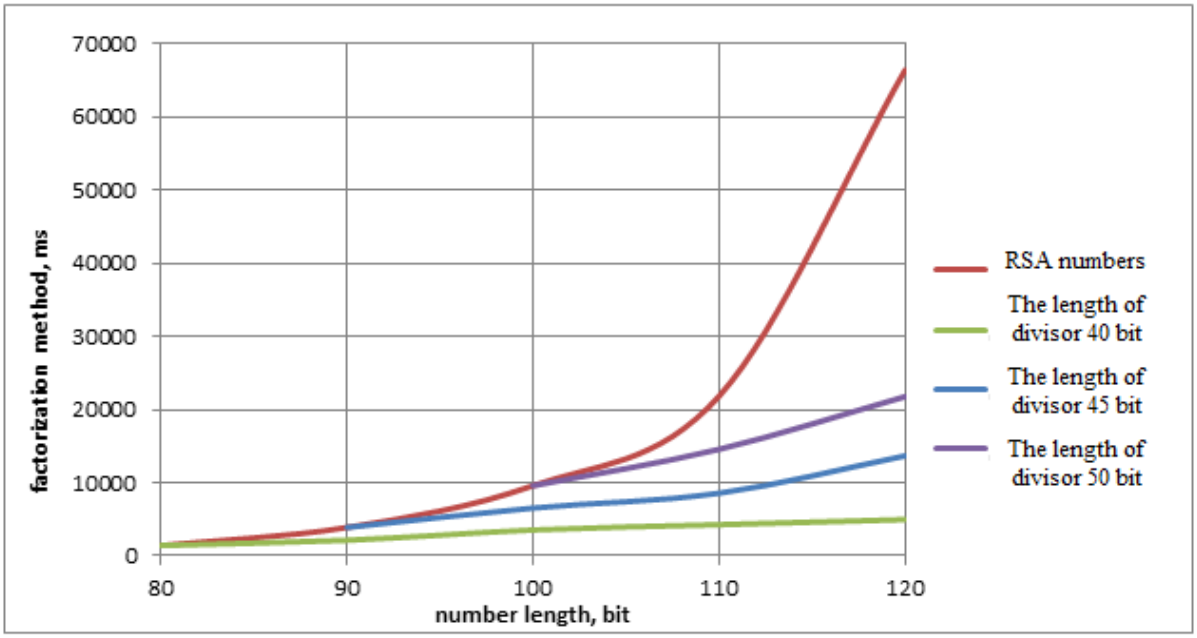

Figure 4. The Speed of Working of the Montgomery Method for the Numbers from 80 to 120 bit.

We can conclude from this graph, that the speed of the algorithm depends not on its length, but on the smallest divisor. Thus, by factoring on elliptic curves one can easily factorize very long numbers the divisors of which do not exceed a certain value. This number did not exceed 25 digitsin this work.

Summary

Summarizing, it is possible to estimate the speed of operation of the four algorithms having been considered. The undisputed leader is the Montgomery method, which among all the studies considered in this article was the fastest and most optimal one. The use of projective coordinates instead of affine coordinates also gives a significant speed increase due to reduction in the number of operations for computing the inverse elements.Comparison between algorithms with one stage and two ones gives preference to the second algorithm, however, this gain could be much larger if the initial boundary *B* could be chosen efficiently. Inefficient choose of *B*can be reduced by using parallelization of the algorithm into several threads. Another reserve for accelerating the algorithm is the possibility of usingthecurves of Edwards [12] to further reduce modular operations.

Acknowledgements

The work is performed according to the Russian Government Program of Competitive Growth of Kazan Federal University.

References

1. Koblitz N., Introduction to elliptic curves and modular forms, Graduate Texts in Mathematics, vol. 97, Springer-Verlag, New York, 1984.
2. Miller V., Use of elliptic curves in cryptography, Advances in cryptology---CRYPTO 85, Springer Lecture Notes in Computer Science vol. 218, 1985, p. 418-425
3. Koblitz N., Menezes A., Vanstone S. The State Of Elliptic Curve Cryptography/ Kluwer Academic Publishers, Boston, 2000, p. 104-105
4. Washington L. Elliptic Curves Number Theory and Cryptography/ L. Washington. – Series Discrete Mathematics and Its Applications, Chapman &Hall/ CRC, second ed. (2008).
5. Blake I.F., Seroussi G., Smart N.P. Advances in Elliptic Curve Cryptography // Cambridge Univ.Press, London Mathematical Society Lecture Note Series. 317 (2005)
6. Lenstra H.W. Factoring integers with elliptic curves/H.W.Lenstra.–Ann.Math. v.126 (1987), p. 649–674
7. Ishmukhametov S.T. TheMethods of Factorization of Natural Numbers /S.T.Ishmukhametov. – Kazan, 2011, p. 89
8. Ismail Amer, ShamilT.Ishmukhametov, Ramilya G. Rubtsova. Lenstra Factorization Method Convergence

Investigation on Elliptic Curves//Research Journal of Applied Sciences 10 (8), 358-364, 2015
9. Crandall R. and Pomerance C. The Prime Numbers: A Computational Perspective (Springer–Verlag, Berlin, New York, 2005), 2-nd ed.
10. Brent R.P. Some integer factorization algorithms using elliptic curves/R.P. Brent.– Austral.Comput.Sci.Comm, 1986, v. 8, p. 149–163
11. Brent R.P., Crandall R.E., Dilcher K., Van Halewyn C. Three New Factors Of Fermat Numbers / Mathematics Of Computation Volume 69, Number 231, 2000, Pages 1297–1304
12. Edwards H.M. A normal form for elliptic curves./ H.M. Edwards.–Bull. Amer. Math. Soc. 44 (2007), p. 393-422

