The Method of Factorizing Multi-Digit Numbers Based on the Operation of Adding Odd Numbers

Mykhailo Kasianchuk[1], Igor Yakymenko[1], Stepan Ivasiev[2], Ruslan Shevchuk[3], Lidiya Tymoshenko[4]

1. Department of Computer Engineering, Ternopil National Economic University, UKRAINE, Ternopil, 8 Chekhova str., email: kasyanchuk@ukr.net, iyakymenko@ukr.net,
2. Department of Cyber Security, Ternopil National Economic University, UKRAINE, Ternopil, 8 Chekhova str., email:stepan.ivasiev@gmail.com
3. Department of Computer Science, Ternopil National Economic University, UKRAINE, Ternopil, 8 Chekhova str., email: rsh@tneu.edu.ua
4. Department of Computer Science and Information Systems Management, Odesa National Polytechnic University, UKRAINE, Odesa, 1 Shevchenko Str., e-mail: lmt0902@gmail.com

Abstract: **The method for factorizing multi-digit numbers based on the addition of odd numbers is developed in this article. The temporal characteristics of software implementations of the proposed method, the classical Fermat method and its improvement are experimentally investigated. It is revealed that the proposed algorithm, whose time reduction is linear, has an advantage over the other two with a large difference between the multipliers that are factorized. Other dependencies are declining parabolic.**

Keywords: **factorization, Fermat method, addition, multi-digit numbers, time complexity.**

I. Introduction

The factorization of a natural number or its decomposition into simple multipliers belongs to the main problems of the modern theory of numbers [1-3], various forms of the system of residual classes [4, 5] and plays an important role in cryptographic information security systems [6-8]. As a rule, large-scale computing and time resources are required for the expansion of the multiplicity of the number obtained as a result of the product of two large prime numbers [9].

This circumstance is used in cryptographic algorithms as protection against potential hacking attempts: for the creation and multiplication of two prime numbers of large bits, it does not require significant costs [10], and their factorization is a long and laborious process. Despite the considerable efforts of scientists in this direction, to date, there are no quantum algorithms for factorizing multi-digit numbers, which allow executing the timetable for the multipliers at a practical time. But the proof that there is no solution to this problem in polynomial time is also absent [11].

Particular interest to the factorization problem was the emergence of cryptographic algorithm encryption RSA, which uses its computational complexity [12].

The decomposition of numbers containing in a binary record up to 768 bits inclusive was obtained for polynomial time by the modern methods (in particular, the general method of the numerical field sieve) [13]. However, further increase in the dimension significantly increases the complexity of the scheduling operation of a composite number. In connection with this, it is necessary to develop other approaches to solving the problem of factorization of integers.

II. Fermat's Factorization Method

Fermat's method is the most widely used method [11], which is based on the search for pairs of natural numbers A and B, for which equality $n=A^2-B^2$ is performed, where $n=p\cdot q$ is known integer, which is the product of two unknown primes p and q, which should be found.

For this purpose $m=\left\lfloor\sqrt{n}\right\rfloor$ is searched and then the parameter $f(x)=(m+x)^2-n$ is calculated, where x=1, 2, 3, … , as long as some value $f(x)$ will not be equal to the full square of a number, for example B^2.

Then, respectively, $A^2=(m+x)^2$ and the desired decomposition will be $n=p\cdot q= A^2-B^2=(A-B)(A+B)$.

The most computable complex operations in this case are elevation to a square and the search for a square root. In [14], an improved Fermat's factorization method is proposed, where the condition is used that squares of integers can be represented as a sum of odd numbers, the number of which is equal to this number:

$$s^2=\sum_{i=1}^{s}(2i-1). \quad (1)$$

Therefore, having found m та $f_1(x)=f(x)$ when x=1, the following steps occur according to the expression $f_i=f_{i-1}+2(m+i)-1$, where i=2, 3, 4, … as long as f_i will not be a full square of a certain number.

The decomposition for multipliers will be determined by this expression: $n=(m+i-f_i)(m+i+f_i)$.

Table 1 provides an example of factorization using the classical and improved Fermat's method for n=4717 ($m_1=\left\lfloor\sqrt{4717}\right\rfloor=68$).

TABLE 1. Example of the factorization of the number 4717 using classical and improved Fermat's method

x	$m+x$	$f(x)$, classical method	$f(x)$, improved method
1	62	$69^2-4717=44$	$62^2-4717=44$
2	63	$70^2-4717=183$	44+139=183
3	64	$71^2-4717=324=18^2$	183+141=324=18^2

Thus the decomposition of the number 4717 on simple multipliers is obtained:

$$4717=71^2-18^2=(71+18)(71-18)=89\cdot 53.$$

It should be noted that the number of iterations in both methods is the same. However, in the improved Fermat's method, the operation of elevating to a square of large numbers is excluded. In addition, arithmetic operations are performed over numbers of a much smaller digit than the classical ones. But the most computationally labor-intensive operation is the extraction of a square root.

Therefore, the purpose of our work is to develop a factorization algorithm based on the Fermat's method, in which there will be no square root search operation, which is especially important for work with numbers of large bit rate for the experimental study, as well as an experimental study of the time characteristics of the software implementation of the Fermat's method, its modification and the proposed algorithm.

III. Description of the Proposed Factorization Algorithm

The property (1) is also used in the proposed algorithm, the parameters $m_1=m$ and $f_{11}=f_1$ are calculated similarly to the previous case. Then the sequence of operations $f_{1i}=f_{1\,i-1}-r_{1\,i-1}$, $r_{11}=1$, $r_{1\,i-1}=2i-3=r_{1\,i-2}+2$, $i=2, 3, \ldots$ are performed until the condition for some i is not fulfilled:

$$f_{1i}-r_{1i}\le 0. \quad (2)$$

Further calculations occur in such a way when performing a strict inequality (2): $m_2=m_1+1$; $f_{21}=f_{1i}+2\ m_2+1-r_{1i}$, $r_{21}=r_{1i}+2$. Searching for the following values f_{2i}, r_{2i} is carried out in the same way as the previous case. In the general case, these calculations can be described by the following expressions:

$$f_{j\,i+1}=f_{ji}-r_{ji}\le 0,\ r_{j\,i+1}=r_{ji}+2, \text{ if } f_{ji}-r_{ji}>0; \quad (3)$$

$$f_{j+1\,1}=f_{ji}+2m_j+1-r_{ji}\le 0,\ r_{j\,i+1}=r_{ji}+2, \text{ if } f_{ji}-r_{ji}<0. \quad (4)$$

Under the condition $f_{ji}-r_{ji}=0$, the desired quantities $A=(m+j)$ and $B=\sqrt{(m+j)^2-n}$ are determined which will be a positive integer, the desired quantity, which will be a natural number.

It should be noted that in this case the value of the parameter j corresponds to the number of steps in the classical and improved Fermat's method.

Fig. 1 shows a block diagram of the developed algorithm, and in the Table 2, the corresponding example is presented for $n=53\cdot 89=4717$ ($m_1=\lceil\sqrt{4717}\rceil=68$).

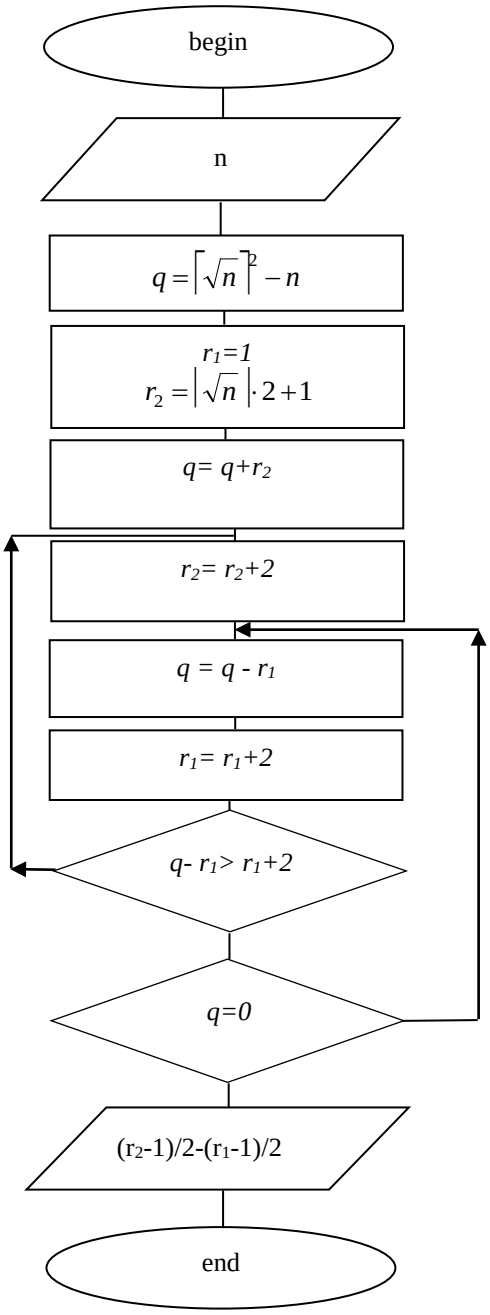

Fig. 1. Block diagram of the developed algorithm

TABLE 2. The example of factorization of the number 4717 by the proposed algorithm

i	$j=1$, $m_1=68$		$j=2$, $m_2=69$		$j=3$, $m_3=70$	
	f_{1i} / r_{1i}		f_{2i} / r_{2i}		f_{3i} / r_{3i}	
1	$69^2-4717=44$	1	$8+139-13=134$	15	$14+141-27=128$	29
2	$44-1=43$	3	$134-15=119$	17	$128-29=99$	31
3	$43-3=40$	5	$119-17=102$	19	$99-31=68$	33
4	$40-5=35$	7	$102-19=83$	21	$68-33=35$	35
5	$35-7=28$	9	$83-21=62$	23	$35-35=0$	
6	$28-9=19$	11	$62-23=39$	25		
7	$19-11=8<13$	13	$39-25=14<27$	27		

Consequently, the decomposition of the number 4717 into simple multipliers are carried out in this way: $B=\sqrt{(68+3)^2-4717}=18$, $4717=71^2-18^2=(71+18)\cdot(71-18)=89\cdot53$. This procedure is performed without the use of a computationally cumbersome operation extracting the square root. Additionally, addition and subtraction are performed over smaller numbers than in the two previous methods, although the number of these operations is greater.

IV. Experimental Study of Time Characteristics of Factorization Methods

The multiplier p was chosen to be fixed and equal to the largest prime number of a certain bit rate for the experimental study of the factorization of the number $n=p\cdot q$.

Then, a number of prime numbers of the same digit are determined that of p, of which 1000 values for the number q were selected uniformly in the order of growth.

After multiplying p by q, the timer fixed the factorization time for each of the 1000 received products.

The time characteristics of the software implementation of the classical (curve 1) and improved (curve 2) Fermat's methods, as well as the proposed algorithm (curve 3) are presented in Figures 2 and 3 for the bit rates 30 (p=1073741789) and 32 (p=4294967291) bits respectively.

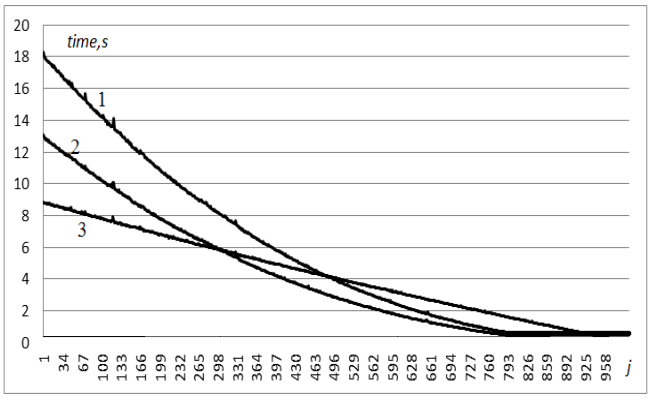

Fig. 2. Time characteristics of software implementation of factorization methods for multipliers with bit rate 30 bits (j – number in the sample)

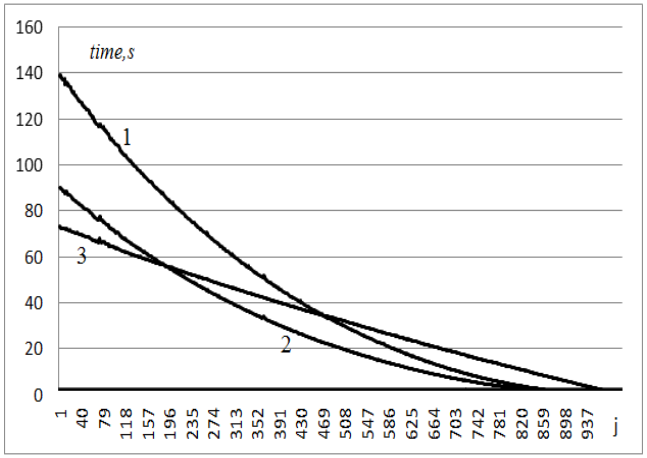

Fig. 3. Time characteristics of software implementation of factorization methods for multipliers with bit rate 32 bits (j – number in the sample)

All calculations were carried out on a portable computer Lenovo B50-70 with processor Intel Pentium 3558U (1.7 GHz). The amount of RAM in the device was 4 GB. When designing the computing software, a high level programming language C ++ was selected, which allows you to transform the codes into different architectures and operating systems.

All graphics are descending character, indicating a decrease in the factorization time with a decrease in the difference between the multipliers, whose product is factorized. The proposed algorithm, whose time reduction is linear, has an advantage over the other two at small values of q. With increasing q the least time is characterized by an improved Fermat's algorithm.

Further growth of q leads to the fact that the proposed algorithm uses the most time compared to the other two. It should be noted that with increasing the bit rate number of the point of intersection of the straight line with the curves obtained by using the Fermat's method, are shifted to the left on the graphs.

Figure 4 depicts the graphs of the dependence of the average factorization time of three methods on the bit rate number for 1000 values selected by the above-described method. It can be seen that all graphs grow in parabolic law with increasing bit rate. The least average factorization time is characterized by an improved Fermat's algorithm, and the largest one is classical.

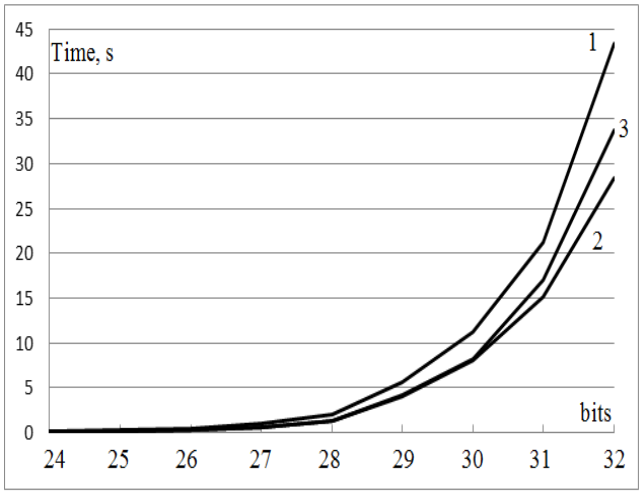

Fig. 4. Graphs of the dependence of the average factorization time on the number of digits: 1 – the classical Fermat's method; 2 – an improved Fermat's method; 3 – the proposed algorithm

Consequently, it is advisable to apply the proposed method when there is a big difference between the numbers whose product is to be factorized.

The Table 3 shows the results of the factorization of the product of two prime numbers for a fixed p=3571 and a variable q (t_1 - the classical Fermat's method, t_2 – the improved Fermat's method, t_3 - the proposed algorithm).

It can be seen from the Table 3 that the improved Fermat's method increases the speed factorization approximately into 1.1-1.2 times, and the proposed method – into 5-11 times compared with the classical Fermat's method.

TABLE 3. INVESTIGATION OF THE TIME CHARACTERISTICS

p	q	$p \cdot q$	t_1, s	t_2, s	t_3, s
3571	99991	357067861	0,031	0,031	0,016
3571	323789	1156250519	0,172	0,157	0,032
3571	523771	1870386241	0,297	0,281	0,047
3571	723791	2584657661	0,438	0,406	0,063
3571	1913803	6834190513	1,156	1,109	0,203
3571	2913803	10405190513	1,781	1,703	0,313
3571	7913809	28260211939	4,86	4,656	0,875
3571	9913807	35402204797	6,265	5,828	1,094
3571	19191383	68532428693	12,843	12,156	2,407
3571	39192331	139955814001	26,875	26,235	4,969
3571	49392341	176380049711	32,859	34,406	6,172
3571	139392347	497770071137	103,078	86,391	18,031
3571	337392373	1204828163983	214,141	220,375	42,483
3571	931392317	3326001964007	646	571,828	116,453
3571	1931392319	6897001971149	1112,657	1106,469	269,156
3571	2971215073	10610209025683	2775,718	2765,922	437,531
3571	6712170737	23969161701827	11159,56	9080,454	937,67

V. CONCLUSIONS

This paper is devoted to the experimental study of the time complexity for the factorization of many digit numbers using the classical and improved Fermat method, and also the factorization algorithm based on the subtraction operation is proposed and investigated. Appropriate graphic dependencies have been constructed. It is shown that the proposed algorithm is characterized by less temporal complexity in the case where the big difference between the numbers whose product must be factorized.

REFERENCES

[1] D. Venturi “Lecture Notes on Algorithmic Number Theory” *New-York-Berlin: Springer-Verlag*, 2009. 217 p.

[2] S. Ishmukhametov, A. Boyko and D. Zyatdinov “On an approach to the problem of the factorization of natural numbers”, *Proceedings of High Schools. Mathematics*, 2011, pp. 15–22.

[3] V. Shoup “Computational Introduction to Number Theory and Algebra”, *Cambridge University Press, Sec.Edition*, 2005, 600 p.

[4] M. Kasianchuk, “The Construction of the modified Perfect Form of Residual Classes System Using Factorization”, *Radio Electronics, Computer Science, Control*, 2017, Vol.42, №3, pp. 53-59.

[5] Ya.M. Nykolaychuk, M.M. Kasianchuk and I.Z. Yakymenko “Theoretical Foundations of the Modified Perfect Form of Residue Number System”, *Cybernetics and Systems Analysis*, 2016, V.52, №2, pp. 219-223.

[6] A.V. Agranovsky and R.A. Hadi “Practical cryptography: algorithms and their programming”, *Moscow: Solon-Press,* 2009, 256 p.

[7] O.N. Vasilenko “Theoretical and numerical algorithms in cryptography”, *Moscow: Center for Continuous Mathematical Education*, 2003, 326 p.

[8] N. Koblitz “Course of Number Theory and Cryptography”, *Moscow: TVP*, 2001, 260 p.

[9] R. Crandall, C. Pomerance “Prime Numbers: A Computational Perspective. Chapter 5: Exponential Factoring Algorithms”, *Springer-Verlag: New York*, 2005, pp. 2-6.

[10] D. Kozaczko, S. Ivasiev, I. Yakymenko and M. Kasianchuk “Vector Module Exponential in the Remaining Classes System”, *Proceedings of the 2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS’2015)*, Warsaw, Poland, 2015, pp. 161-163.

[11] S Ishmukhametov “Methods for the factorization of natural numbers: a tutorial”, *Kazan: Kazan University*, 2011, 190 p.

[12] E.B. Makhovenko “Theoretical and numerical methods in cryptography: Textbook”, *Moscow: Helios ARV*, 2006, 320 p.

[13] RSA Challenge. URL: https://www.emc.com/emc-%20plus/rsa-labs/historical/the-rsa-challenge-numbers.htm.

[14] M. Karpiński, S. Ivasiev, I. Yakymenko, M. Kasianchuk and T. Gancarczyk “Advanced method of factorization of multi-bit numbers based on Fermat's theorem in the system of residual classes”, *Proc. of 16th International Conference on Control, Automation and Systems (ICCAS–2016)*, Gyeongju, Korea, V.1, October, 2016, pp. 1484–1486.