The difficulty of prime factorization is a consequence of the positional numeral system

Yaroslav D. Sergeyev*

Abstract

The importance of the prime factorization problem is very well known (e.g., many security protocols are based on the impossibility of a fast factorization of integers on traditional computers). It is necessary from a number k to establish two primes a and b giving $k = a \cdot b$. Usually, k is written in a positional numeral system. However, there exists a variety of numeral systems that can be used to represent numbers. Is it true that the prime factorization is difficult in any numeral system? In this paper, a numeral system with partial carrying is described. It is shown that this system contains numerals allowing one to reduce the problem of prime factorization to solving $[K/2] - 1$ systems of equations, where K is the number of digits in k (the concept of digit in this system is more complex than the traditional one) and $[u]$ is the integer part of u. Thus, it is shown that the difficulty of prime factorization is not in the problem itself but in the fact that the positional numeral system is used traditionally to represent numbers participating in the prime factorization. Obviously, this does not mean that P=NP since it is not known whether it is possible to re-write a number given in the traditional positional numeral system to the new one in a polynomial time.

Key Words: Numeral systems, partial carrying, prime factorization.

1 Introduction

In the prime factorization problem it is necessary from a number k written in a positional numeral system to reestablish two primes a and b such that $k = a \cdot b$. The importance of this problem is very well known. For instance, many security protocols are based on the impossibility of a fast factorization of integers on traditional computers. Numerous algorithms working with conventional and unconventional paradigms have been proposed for this purpose (see, e.g., [1,3,6,25] and references given therein). In particular, the result of Shor (see [25]) showing that this problem can be efficiently solved on a quantum computer has attracted a great attention not

*Distinguished Professor, University of Calabria, Italy; he works also at the Lobachevsky State University of Nizhni Novgorod, Russia and the Institute of High Performance Computing and Networking of the National Research Council of Italy, yaro@dimes.unical.it http://wwwinfo.dimes.unical.it/∼yaro

only of specialists but also of a general public (on October 2016 Google Scholar reports almost 6000 citations to his paper [25]).

Traditionally it is tacitly supposed that a number to be factorized is written in a positional numeral system, and prime factors to be found should be written in the same system, as well. This paper separates the problem of factorization of a number from the problem of representation of the number. What will happen if the positional system is substituted with another numeral system? Will the factorization problem remain as difficult as it is with the usual positional system? In fact, there exists a variety of other numeral systems that can be used to represent numbers: Egyptian system, Roman system, unary system, etc. (see, e.g., [4] for numerous examples of numeral systems). It is well known that operations with some numerals are more difficult than with others. For instance, division with Roman numerals is rather tricky (see, e.g., [8]).

In order to start the discussion it is obligatory first to consider the positional numeral system. Such a system with a finite integer radix r uses symbols $\{0, 1, \ldots, r-1\}$ called *digits*. In this system, an integer a is expressed by the record

$$(a_n a_{n-1} \ldots a_1 a_0)_r, \tag{1}$$

where all digits a_i satisfy the condition $0 \leq a_i \leq r-1,\ 0 \leq i \leq n$, and the numeral (1) represents the quantity calculated as follows

$$a = a_n r^n + a_{n-1} r^{n-1} + \ldots + a_1 r^1 + a_0 r^0. \tag{2}$$

Thus, the quantity that represents by each digit a_i depends on its position in the numeral (1). The finiteness of r and the fact that each digit a_i satisfies the condition $0 \leq a_i \leq r-1,\ 0 \leq i \leq n$, leads to the necessity to carry forward digits to the position $i+1$ if a result of an arithmetic operation in a position i is larger than $r-1$.

Notice that there exist mathematical objects such that operations of multiplication and addition with these objects are executed without carrying. The first example is obviously polynomials. Let us consider, for instance, the following multiplication

$$(a_2 x^2 + a_1 x^1 + a_0) \cdot (b_2 x^2 + b_1 x^1 + b_0) = k_4 x^4 + k_3 x^3 + k_2 x^2 + k_1 x^1 + k_0. \tag{3}$$

Carrying is not present here since the value of x is not defined and coefficients of the resulting polynomial are calculated as follows

$$k_4 = a_2 b_2, \qquad k_3 = a_2 b_1 + a_1 b_2, \tag{4}$$

$$k_2 = a_2 b_0 + a_1 b_1 + a_0 b_2, \qquad k_1 = a_1 b_0 + a_0 b_1, \qquad k_0 = a_0 b_0. \tag{5}$$

The second example of addition and multiplication without carrying is a recently introduced numeral system allowing one to execute numerical computations with infinite and infinitesimal numbers (see [9,14,16,21,22] and the patent [18] describing the Infinity Computer able to work with this kind of numbers). In this numeral system, the radix b is infinite and all digits a_i are finite. Thus, in practical computations where the number of operands is finite it cannot happen that multiplication or addition of two digits is infinite and so there is no necessity for carrying. Computations of this kind have shown to be very efficient in a number of applications (see, e.g., [5, 7, 10, 13, 15, 17, 19, 20, 23, 24, 26, 27]).

In this paper, the problem of prime factorization where numbers k, a, and b are represented in the positional numeral system (1), (2) is not discussed. A positional numeral system with partial carrying is described (Section 2) and it is shown (Section 3) that prime factorization in this system is much easier than the same problem with numbers expressed in the system (1), (2). Examples are given in Section 4 and Section 5 concludes the paper. Notice that the results obtained in this paper do not imply that P=NP since it is not known whether it is possible to re-write a number given in the traditional positional numeral system to the new one in a polynomial time.

2 *A positional numeral system with partial carrying*

In the numeral system with partial carrying called P_r hereinafter prime numbers are written in the usual positional way with the base r as in (1), (2) where r is a positive integer. All other numerals are constructed as results of operations with primes following the rules of operations with polynomials (see, e.g., [2] for detailed discussions on work with polynomials), i.e., without carrying between powers of x, where it is taken $x = r$. Let us show a way to construct numerals in P_r using multiplication (that will be of a principal interest in this paper) and following an analogy with multiplication of polynomials. A numeral obtained as the result of multiplication of two prime numbers is called the *generative form* hereinafter.

We consider two prime numbers a and b and their product $k = a \cdot b$ written in the system P_r in the generative form, i.e., as the result of multiplication of a and b with partial carrying, where

$$a = (a_n a_{n-1} \dots a_1 a_0)_r = a_n r^n + a_{n-1} r^{n-1} + \dots + a_1 r^1 + a_0 r^0, \qquad (6)$$

$$b = (b_m b_{m-1} \dots b_1 b_0)_r = b_m r^m + b_{m-1} r^{m-1} + \dots + b_1 r^1 + b_0 r^0, \qquad (7)$$

$$0 \leq a_i \leq r-1, \quad 0 \leq i \leq n, \quad 0 \leq b_j \leq r-1, \quad 0 \leq j \leq m. \qquad (8)$$

and the result of multiplication is

$$k = k_{n+m} r^{n+m} + \dots + k_1 r^1 + k_0 r^0, \qquad (9)$$

where numbers $k_i,\ 0 \le i \le n+m$, are calculated as follows

$$\left(\begin{array}{lcl} k_{n+m} & := & a_n b_m, \\ k_{n+m-1} & := & a_n b_{m-1} + a_{n-1} b_m, \\ k_{n+m-2} & := & a_n b_{m-2} + a_{n-1} b_{m-1} + a_{n-2} b_m, \\ & \dots & \\ k_s & := & \sum\limits_{\substack{j+j=s \\ 0 \le i \le n \\ 0 \le j \le m}} a_i b_j \\ & \dots & \\ k_2 & := & a_0 b_2 + a_1 b_1 + a_2 b_0, \\ k_1 & := & a_0 b_1 + a_1 b_0, \\ k_0 & := & a_0 b_0, \end{array}\right). \quad (10)$$

In the numeral system P_r, since it works following the rules of polynomials, conditions analogous to (8) can be broken with respect to k during multiplication as far as there is no carrying between powers of $x = r$. In fact, there can be numbers k_i such that $k_i > r$ and to express them in the positional system with the radix r more than one symbol can be required, i.e., it can be that

$$k_i > r, \qquad k_i = (k_{i,j} k_{i,j-1} \dots k_{i,0})_r, \qquad 0 \le k_{i,l} \le r-1, \quad 0 \le l \le j. \quad (11)$$

In case condition (11) holds for k_i, it is called *compound digit*, whereas digits k_j consisting of one symbol only and satisfying condition $0 \le k_j \le r-1$ are called *simple digits*. The product k has so $K+1$ digits $k_i,\ 0 \le i \le K$, where $K = n+m$ and some digits can be simple and other compound. The numeral system P_r is called "with partial carrying" since the carrying is present inside compound digits but is not executed between digits k_i and $k_{i+1},\ 0 \le i < K$. In order to separate compound and simple digits some signs should be used. For instance, compound digits can be bounded by signs | or underlined in a way. Let us show hot to do it by the following example.

We wish to multiply two prime numbers a and b using P_r and to compute $k = a \cdot b$ where

$$a = (a_2 a_1 a_0)_r, \qquad b = (b_2 b_1 b_0)_r, \quad (12)$$

digits of a and b satisfy (8), and k is calculated following (10). Since for both a and b we have $n = m = 2$, it follows from (10) that $K = 2+2 = 4$ and the product k will contain 5, digits: k_4, k_3, k_2, k_1, and k_0. If we suppose that digits k_4, k_2, and k_0 are simple and digits k_3 and k_1 are compound such that $k_3 = (k_{3,2} k_{3,1} k_{3,0})_r$ and $k_1 = (k_{1,1} k_{1,0})_r$, then the number k can be written either as

$$k = (k_4 | k_{3,2} k_{3,1} k_{3,0} | k_2 | k_{1,1} k_{1,0} | k_0)_r$$

or as

$$k = (k_4 \underbracket{k_{3,2} k_{3,1} k_{3,0}} k_2 \underbracket{k_{1,1} k_{1,0}} k_0)_r.$$

Let us give a numerical example (the first way to separate compound digits is used hereinafter). Suppose that we wish to multiply prime numbers $a = 127$ and $b =$

359 written using $r = 10$. Then in the traditional decimal system we have $k = 127 \cdot 359 = 45993$. In the numeral system P_{10}, by applying formulae (3)–(5) we get

$$k = 1{\cdot}3{\cdot}10^4+(1{\cdot}5+2{\cdot}3){\cdot}10^3+(1{\cdot}9+2{\cdot}5+7{\cdot}3){\cdot}10^2+(2{\cdot}9+7{\cdot}5){\cdot}10^1+(7{\cdot}9){\cdot}10^0 =$$

$$3 \cdot 10^4 + 11 \cdot 10^3 + 40 \cdot 10^2 + 53 \cdot 10^1 + 63 \cdot 10^0 = 3|11||40||53||63|. \quad (13)$$

It should be emphasized that P_r allows multiple representations of integers. Together with the generative way of constructing new numerals other representations are also possible as results of addition, subtraction, and division. For instance, the quantity represented by the numeral k from (13) can be obtained as the result of addition

$$\underbrace{1+1+1\ldots+1+1}_{45993 \text{ addends}} = |45993|$$

giving us only one compound digit equal to 45933. It can be represented also as numerals obtained by adding lacking quantities to generative numerals obtained by multiplication of two primes less than $a = 127$ and $b = 359$, etc.

Notice also that, as it happens for polynomials where their coefficients can be positive or negative, subtraction in P_r can lead to numerals having negative (compound or simple) digits. Signed-digit representations are well known (see, e.g., [11, 12] and references given therein for discussions on advantages and disadvantages of several kinds of these systems).

3 *Generative representation and prime factorization*

Let us consider now the problem of prime factorization using numerals written in P_r. Suppose that there are two prime numbers a and b from (6), (7) and their product $k = a \cdot b$ from (9) is written in the system P_r in the generative form, i.e., as the result of multiplication of a and b with partial carrying calculated following (10). Suppose also that numbers n and m are known. Our goal is to reestablish numbers a and b from k.

To do so let us consider (10) as a system of $n+m+1$ equations with $n+m+2$ unknowns $a_i,\ 0 \le i \le n$, and $b_j,\ 0 \le j \le m$, and the known numbers $k_i,\ 0 \le i \le n+m$.

$$\left\{ \begin{array}{lcl} a_n b_m & = & k_{n+m}, \\ a_n b_{m-1} + a_{n-1} b_m & = & k_{n+m-1} \\ a_n b_{m-2} + a_{n-1} b_{m-1} + a_{n-2} b_m & = & k_{n+m-2} \\ b_{m-3} + a_{n-1} b_{m-2} + a_{n-2} b_{m-1} + a_{n-3} & = & k_{n+m-3} \\ \ldots & & \\ b_3 + a_1 b_2 + a_2 b_1 + a_3 & = & k_3 \\ a_0 b_2 + a_1 b_1 + a_2 b_0 & = & k_2 \\ a_0 b_1 + a_1 b_0 & = & k_1 \\ a_0 b_0 & = & k_0, \end{array} \right. \quad (14)$$

The following theorem then follows.

Theorem 1 *Suppose that primes a and b and their product $k = a \cdot b$ are written in the system P_2 in the generative form and numbers n and m from (6), (7) are known, then system (14) can be reduced to a system with $n + m - 1$ equations and $n + m - 2$ unknowns giving so the possibility to find a and b.*

Proof. Since the product k is written in the generative form, numbers $k_i,\ 0 \leq i \leq n + m$, are known. Then, due to the fact that $r = 2$ and n and m are known, it should be $a_n = b_m = 1$. Numbers a and b are primes, therefore, they are odd and this implies that $a_0 = b_0 = 1$. Thus, (14) re-written using the established values $a_n = b_m = a_0 = b_0 = 1$ (and with the first and the last lines omitted since they are identities) becomes

$$\begin{cases} b_{m-1} + a_{n-1} & = & k_{n+m-1} \\ b_{m-2} + a_{n-1}b_{m-1} + a_{n-2} & = & k_{n+m-2} \\ b_{m-3} + a_{n-1}b_{m-2} + a_{n-2}b_{m-1} + a_{n-3} & = & k_{n+m-3} \\ \dots & & \\ b_3 + a_1b_2 + a_2b_1 + a_3 & = & k_3 \\ b_2 + a_1b_1 + a_2 & = & k_2 \\ b_1 + a_1 & = & k_1 \end{cases} \tag{15}$$

This system contains $n + m - 1$ equations and $n + m - 2$ unknowns. □

Suppose now that we know only the value $K = n + m$ but numbers n and m are unknown. Then the following Corollary holds.

Corollary 1 *Given the number k, $[K/2] - 1$ systems of the kind (15) should be solved to find a and b, where $[u]$ is the integer part of u. One of these systems gives the solution and the others are inconsistent.*

Proof. The proof is straightforward. Without loss of generality let us suppose that $n < m$ in (15). Since the numeral k consists of $K + 1$ digits and in P_2 the smallest prime consists of 2 digits ($(3)_{10} = (11)_2$) then the possible pairs of n and m giving $K + 1$ digits in k are: $(2, K - 2), (3, K - 3), \dots (l, g)$, where $l = g = K/2$ in case K is even and $l = g - 1 = [K/2]$ if K is odd. Since numbers a and b are prime, only one of these systems gives the solution and the other systems are inconsistent. □

4 Examples of factorization

Suppose that we have a number k being the product of two primes, written in the generative form using the numeral system P_2, and consisting of $K+1$ digits, simple and/or compound. Then, as it has been established in the previous section, in order to find its prime factors a and b, it is necessary to solve at maximum $[K/2] - 1$ systems (15). Notice that, since all digits a_i and all b_j should be nonnegative, equations having $k_i = 0$ help a lot to solve (15). Namely, they either allow us to determine that several digits in a and b are equal to zero simplifying so (15) or to establish that the system (15) under consideration is inconsistent.

Let us consider as an example multiplication with $a = 13$ and $b = 19$ giving $a \cdot b = k = 247$. In the system P_2 we obtain

$$a \cdot b = 1101 \cdot 10011 = k = 110|10||10|111 =$$
$$1 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 10 \cdot 2^4 + 10 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0. \qquad (16)$$

Thus, the problem consists of finding a and b from $k = 110|10||10|111$. This numeral has $K = 7$ and it consists of 8 digits

$$k_7 = 1, \quad k_6 = 1, \quad k_5 = 0, \quad k_4 = 10, \quad k_3 = 10, \quad k_2 = 1, \quad k_1 = 1, \quad k_0 = 1,$$

where k_4 and k_3 are compound and the remaining digits are simple.

Since $K = 7$, it follows from Corollary 1 that we should consider two systems: $n = 2, m = 5$ and $n = 3, m = 4$. Let us consider the first pair, $n = 2, m = 5$, i.e., we test the hypothesis that multipliers a and b have the following form

$$a = a_2 a_1 a_0, \qquad b = b_5 b_4 b_3 b_2 b_1 b_0.$$

Taking into account that $a_2 = b_5 = a_0 = b_0 = 1$ system (15) becomes

$$\left\{ \begin{array}{lcl} a_1 + b_4 & = & 1 \\ b_3 + 1 + a_1 b_4 & = & 0 \\ a_1 b_3 + b_2 + b_4 & = & 10 \\ b_1 + b_3 + a_1 b_2 & = & 10 \\ a_1 b_1 + 1 + b_2 & = & 1 \\ b_1 + a_1 & = & 1 \end{array} \right. \qquad (17)$$

Since all a_i and all b_j should be nonnegative, it follows immediately from the second equation of (17) that this system has no solution.

Let us consider now the second hypothesis, i.e., that multipliers have the form

$$a = a_3 a_2 a_1 a_0, \qquad b = b_4 b_3 b_2 b_1 b_0.$$

Since in this case $n = 3$ and $m = 4$ and $a_3 = b_4 = a_0 = b_0 = 1$, system (15) becomes

$$\left\{ \begin{array}{lcl} b_3 + a_2 & = & 1 \\ b_2 + a_2 b_3 + a_1 & = & 0 \\ b_1 + a_2 b_2 + a_1 b_3 + 1 & = & 10 \\ b_3 + a_1 b_2 + a_2 b_1 + 1 & = & 10 \\ b_2 + a_1 b_1 + a_2 & = & 1 \\ b_1 + a_1 & = & 1 \end{array} \right. \qquad (18)$$

This system having six equations and five unknowns can be easily solved. As it was in the previous case, equations with the right-hand part equal to zero show to be very useful. In this case regarding system (18), the second equation of (18) allows us to conclude immediately that $b_2 = a_2 b_3 = a_1 = 0$. Then, the last equation gives us $b_1 = 1$, the penultimate one provides $a_2 = 1$, and the first one $b_3 = 0$ completing so the reconstruction of the factors a and b from (16).

5 A brief conclusion

In this paper, the prime factorization problem was considered from the point of view of numeral systems used to represent integers. The question that was investigated was: 'Is it true that the prime factorization is difficult in any numeral system?'

A numeral system with partial carrying has been described. The concept of digit in this system is more complex than the traditional one, namely, each digit can consist of several symbols. It was shown that this system contains numerals allowing one to reduce the problem of prime factorization to solving $[K/2] - 1$ systems of binary equations, where K is the number of digits in the number k that should be factorized and $[u]$ is the integer part of u. One of these systems of binary equations gives the solution and the others are inconsistent. A method allowing one to solve the systems easily has been suggested. Examples of factorization were given.

Thus, the numeral system P_2 with partial carrying and the information that the numeral k written in this system has been obtained as a result of multiplication of primes a and b give the possibility to find the two multipliers a and b. This means, that the difficulty of the problem of prime factorization is due to the usage of the traditional positional system for representing numbers and not due to the problem of factorization itself. It should be stressed that the obtained results cannot be considered as a proof of P=NP, since it is not known whether it is possible to rewrite a number given in the traditional positional numeral system to the new one in a polynomial time.

References

[1] A. Adamatzky. *Physarum Machines: Computers from Slime Mould*. Series A 74. World Scientific Series on Nonlinear Science.

[2] E. J. Barbeau. *Polynomials*. Springer, New York, 2 edition, 2003.

[3] E. Blakey. Towards non-quantum implementations of Shor's factorization algorithm. *Int. Journ. of Unconventional Computing*, 10:339–352, 2014.

[4] L. Corry. *Numbers*. Oxford University Press, Oxford, 2015.

[5] S. De Cosmis and R. De Leone. The use of grossone in mathematical programming and operations research. *Applied Mathematics and Computation*, 218(16):8029–8038, 2012.

[6] R. Crandall and C. Pomerance. *Prime Numbers: A Computational Perspective*. Springer, New York, 2001.

[7] L. D'Alotto. A classification of one-dimensional cellular automata using infinite computations. *Applied Mathematics and Computation*, 255:15–24, 2015.

[8] J.G. Kennedy. Arithmetic with Roman numerals. *American Mathematical Monthly*, 88(1):29–32, 1981.

[9] G. Lolli. Metamathematical investigations on the theory of grossone. *Applied Mathematics and Computation*, 255:3–14, 2015.

[10] M. Margenstern. Fibonacci words, hyperbolic tilings and grossone. *Communications in Nonlinear Science and Numerical Simulation*, 21(1–1):3–11, 2015.

[11] B. Parhami. Generalized signed-digit number systems: A unifying framework for redundant number representations. *IEEE Transactions on Computers*, 39(1):89–98, 1990.

[12] J. Rebacz, E. Oruklu, and J. Saniie. Fast signed-digit multi-operand decimal adders. *Circuits and Systems*, 2:225–236, 2011.

[13] D. Rizza. Supertasks and numeral systems. In Sergeyev Ya.D., Kvasov D.E., Dell'Accio F., and Mukhametzhanov M.S., editors, *Proc. of the 2nd Intern. Conf. "Numerical Computations: Theory and Algorithms"*, volume 1776, page 090005. AIP Publishing, New York, 2016.

[14] Ya.D. Sergeyev. *Arithmetic of Infinity*. Edizioni Orizzonti Meridionali, CS, 2003, 2d electronic ed. 2013.

[15] Ya.D. Sergeyev. Blinking fractals and their quantitative analysis using infinite and infinitesimal numbers. *Chaos, Solitons & Fractals*, 33(1):50–75, 2007.

[16] Ya.D. Sergeyev. A new applied approach for executing computations with infinite and infinitesimal quantities. *Informatica*, 19(4):567–596, 2008.

[17] Ya.D. Sergeyev. Evaluating the exact infinitesimal values of area of Sierpinski's carpet and volume of Menger's sponge. *Chaos, Solitons & Fractals*, 42(5):3042–3046, 2009.

[18] Ya.D. Sergeyev. *Computer system for storing infinite, infinitesimal, and finite quantities and executing arithmetical operations with them*. USA patent 7,860,914, 2010.

[19] Ya.D. Sergeyev. Higher order numerical differentiation on the infinity computer. *Optimization Letters*, 5(4):575–585, 2011.

[20] Ya.D. Sergeyev. Solving ordinary differential equations by working with infinitesimals numerically on the infinity computer. *Applied Mathematics and Computation*, 219(22):10668–10681, 2013.

[21] Ya.D. Sergeyev. Computations with grossone-based infinities. In Calude C.S. and Dinneen M.J., editors, *Unconventional Computation and Natural Computation: Proc. of the 14th International Conference UCNC 2015*, volume LNCS 9252, pages 89–106. Springer, New York, 2015.

[22] Ya.D. Sergeyev. Un semplice modo per trattare le grandezze infinite ed infinitesime. *Matematica nella Società e nella Cultura: Rivista della Unione Matematica Italiana*, 8(1):111–147, 2015.

[23] Ya.D. Sergeyev and A. Garro. Observability of Turing machines: A refinement of the theory of computation. *Informatica*, 21(3):425–454, 2010.

[24] Ya.D. Sergeyev, M.S. Mukhametzhanov, F. Mazzia, F. Iavernaro, and P. Amodio. Numerical methods for solving initial value problems on the Infinity Computer. *Int. Journ. of Unconventional Computing*, 12(1):3–23, 2016.

[25] P.W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.

[26] A.A. Zhigljavsky. Computing sums of conditionally convergent and divergent series using the concept of grossone. *Applied Mathematics and Computation*, 218(16):8064–8076, 2012.

[27] A. Žilinskas. On strong homogeneity of two global optimization algorithms based on statistical models of multimodal objective functions. *Applied Mathematics and Computation*, 218(16):8131–8136, 2012.