Digit-Wise Parallelism of Additive Operations in Numeration Systems with Redundant Digits

Nikolai N. Nepejvoda[1[0000-0002-7869-8053]], Ivan N. Grigorevskiy[1[0000-0001-8078-1238]], Andrei V. Klimov[2[0000-0003-0418-7311]], Yuri A. Klimov[2[0000-0001-5081-1547]], Sergei A. Romanenko[2[0000-0002-2971-9647]], and Artem B. Shvorin[1[0000-0002-3555-0905]]

[1] Ailamazyan Program Systems Institute of RAS,
Peter I str., 4a, selo Veskovo, Pereslavl-Zalessky, Yaroslavl region, 152021, Russia
[2] Keldysh Institute of Applied Mathematics of RAS,
Miusskaya sq., 4, Moscow, 125047, Russia
nepejvodann@gmail.com, ivan_ger@mail.ru, klimov@keldysh.ru, yuklimov@keldysh.ru, romansa@keldysh.ru, art@shvorin.net

Abstract. An introduction and the state of work of the project for the implementation of high-precision arithmetic in an overlaying numeration system, which provides digit-wise parallelism of operations of addition and subtraction of real numbers with restricted carry propagation only by 1–2 bits, is presented. The historical origins of ideas behind such numeration systems in the history of mathematics, dating back to A.L. Cauchy (1840) and L.E.J. Brouwer (1921), are outlined. We explain a simplest overlaying system referred to as "three halves". It is built on the binary system by adding a extra (redundant) third digit representing an interval overlaying with the intervals of 0 and 1. We demonstrate an addition scheme for it, which propagates a carry to next two positions. A simpler scheme with carry propagation to only one position is also shown, and we argue that it requires two redundant digits and a numeration system base not less than 3.

Keywords: arbitrary-precision arithmetic, bignum arithmetic, digit-wise parallelism, overlaying numeration system.

1 Introduction

Problem statement. Modern problems of mathematical modeling require an increase in the calculation accuracy and bit width of numbers, some up to thousands or more bits. The digit-wise-parallel implementation of operations on such long numbers faces the carry problem: in the usual column addition scheme, the carry is performed in the worst case in a time proportional to the bit width, while it is intuitively expected that addition can be performed with bit parallelism in one clock cycle. For example, when adding two binary numbers in Figure 1 on the left, the fractional part of the result consists of some units; when replacing the least significant digit of the second number from 0 to 1, there is a carry along the entire mantissa (Figure 1 on the right). There are more

```
  0,10101010101010101010101      0,10101010101010101010101
+ 0,01010101010101010101010  +   0,01010101010101010101011
= 0,11111111111111111111111  =   1,00000000000000000000000
```

Fig. 1. Adding two numbers with a difference in one least significant bit.

complex schemes of carry in logarithmic time [8, Chapter 29], but they have to waste the area of the crystal.

A radical solution could be such a numeration system in which there are no carries at all for an unlimited number of digits. However, as mentioned by L.E.J. Brouwer a hundred years ago [1], unlimited carriers are present in any positional numeration system in which the number of digits coincides with the base of the system.

Brouwer was worried about a fundamental mathematical problem that arose in the development of *intuitionistic mathematics*: how to represent real numbers and other (potentially) infinite mathematical objects in such a way that the *Principle of Continuity* is fulfilled? It is also called the *Principle of Finite information* [10, 12–14]:

Any finite part of the result must be returned over some finite part of the argument(s).

The representation of real numbers is (potentially) infinite towards the lower-order bits. The final part of such a representation is a sequence of bits from the most significant to some least significant. Arithmetic operations, for example addition, in order to satisfy the requirements of intuitionistic mathematics, must return each fragment of the representation from the most significant digits, "consuming" a certain finite number of argument digits. However, as we see in Figure 1, if the addition operation in the binary system produces only ones in a row for the leading parts of the arguments, we do not know if it will reset them all to zero at some future moment. Moreover, as we know from the first grade, addition and multiplication in a column are performed from low-order bits to high-order ones, and Brouwer needed other algorithms of operations for real numbers - from high-order bits to low-order ones, since infinity in their provision goes towards the low-order bits.

We observe an interesting phenomenon: seemingly completely different questions from the opposite areas of theory/practice on the axis of mathematics: the foundations of mathematics, on the one hand, and digit-wise parallelization of arithmetic operations, on the other, have been faced same pitfalls. This means that the problem is worthy of systematically tackling it, but you do not need to expect quick, simple results.

Approaches to the solution. The first solution we discovered (thanks to A.B. Shvorin [15, 16]), close to our work, goes back to the paper by A.L. Cauchy in 1840 [3]. Cauchy noticed that in numeration systems with base 10, but with an extended set of 11 digits from −5 to 5, hyphenations are limited, which can make calculations easier and reduce errors.

To solve his problem of developing arithmetic operations that satisfy the Principle of Continuity Brouwer [1] designed a numeration system with base 2 with three digits. In such a system, it became possible to sequentially receive the highest bits of the result, gradually consuming the bits of the summands, satisfying the Principle of Continuity and Finite Information.

Additional digits lead to redundancy in the representation, when the intervals of the

numeric axis corresponding to the digits intersect, overlay. Therefore, such systems are called overlaying systems. It is thanks to this redundancy and overlaying that it is possible to "extinguish" the unlimited propagation of carry during addition and to achieve digit-wise parallelism only with local interaction of neighboring parallel blocks. Cauchy and Brouwer each used one additional digit over and above what is required for this reason. You can add more numbers, getting some more benefits (one of which we will show below in Figure 6).

We develop these ideas of the classics further in order to bring numeration systems with redundant digits to practice and implementation of circuitry in equipment.

Other and related works. With the advent of the computer era, and especially in recent decades, the development of alternative representations of numbers has become very popular. Many works have been published that cannot be reviewed in a short paper. Among the recent achievements, we note the work of the group led by Christiane Frougny [4, 5], which studied the necessary conditions for the existence of a parallel addition algorithm.

Another direction with a long history is modular arithmetic based on the residual class system (RNS). It is interesting that in the USSR in the 1960s, computers based on modular arithmetic were developed in Zelenograd for use in missile defense systems [9]. According to B.M. Malashevich [9], the "K-340A" computer was the first in the world to overcome the boundary of 1 million op/sec (on numbers with a fixed point in the range of $\pm 1.6\times10^{12}$), and he believes that modular arithmetic gave a 7-fold increase in performance compared to what was achievable on that hardware. From modern works, profound results were obtained by a group led by V.S. Knyazkov [6, 7], who solved a number of theoretical problems and developed algorithms and implementation on graphic coprocessors of all arithmetic operations, including division. Note that division required an approximate estimate of intermediate results, which is calculated using interval arithmetic. The arithmetic discussed in this paper is essentially interval arithmetic. This opens the possibility of its application in modular division.

More detailed historical information about these and other related works can be found in the introduction of paper [15].

Status of work and content of the paper. In recent years, the authors, together with colleagues, have checked the theoretical study of overlaying numeration systems, the properties, general and special cases of such systems have been studied [11, 12–16]. Currently, the development of circuitry solutions based on FPGAs is underway with the further purpose of conducting an experimental study on a number of applied problems and assessing whether the use of such systems is justified.

In this paper:

1. An introduction to overlaying numeration systems and a historical background is given.
2. Digit-wise-parallel schemes of addition with carry of only one and two next digit positions in a system with one and two additional digits are demonstrated. As far as we know, the observation that such a scheme is possible was made by us for the first time. In previous works, the problem of reducing the number of additional digits was often solved, which caused a carry to a larger number of digit positions.

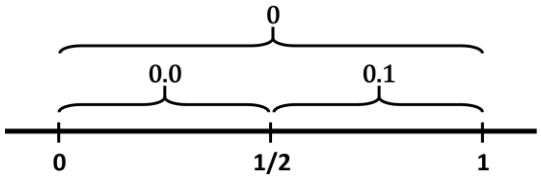

Fig. 2. Intervals corresponding to digits in the usual binary notation.

2 Overlaying Systems

As a starting point for constructing overlaying systems, consider the familiar binary numeration system. Figure 2 shows the interval [0, 1) of the numerical axis. The parenthesis from 0 to 1 at the top indicates the range of numbers that start at 0. Divide the interval in half. Writing numbers from [0,1/2) starts from 0.0; in the interval [1/2,1) – from 0.1, which is shown by the corresponding brackets.

Before moving on to overlaying systems, we stock up on the observation that representations in the positional numeration system, such as 0, 0.0, 0.1, can be considered an image of not one really number standing at the beginning of the interval, but the interval itself, and, having in mind such semantics, to define arithmetic operations that map intervals to intervals. In this case, it is often necessary to give the answer in the form of an upper bound, generalizing the resulting intervals to wider ones, since the exact intervals will turn out to be unimaginable. This interpretation is not used in conventional positional numeration systems. However, in overlaying systems, on the contrary, the standard interpretation as denoting an exact number gives rise to certain problems, and the interval semantics of operations is convenient and understandable.

Now let's look at the simplest overlaying numeration system, which we call "three halves". The base of the system is still 2, that is, it can be considered binary. However, now we will have three digits: in addition to 0 and 1, there is another additional one, which we represent as ½. This number will represent the interval [1/4,3/4). The numbers, as well as the smaller intervals included in it, will get another image of 0.½ in the first two digits. This interval is shown in Figure 3 in parenthesis below the number axis. Note that the left and right halves of this interval are written using three digits in two ways, for example, the right side – in the form of 0.10 and 0.½ (lower bracket in Figure 3).

We have obtained a system in which an alphabet of a larger number of digits is used to represent numbers and intervals of the same precision. This requires additional hardware resources per digit. But it is precisely this redundancy that makes it possible to "extinguish" the carriers, "absorb" them with additional digits. To see this, it is necessary to formally describe the tables of addition and subtraction operations. Some of them are given in paper [14]. Tables of operations for overlaying systems are ambiguous, and the option can be selected from other considerations – aesthetic, practical or circuit design.

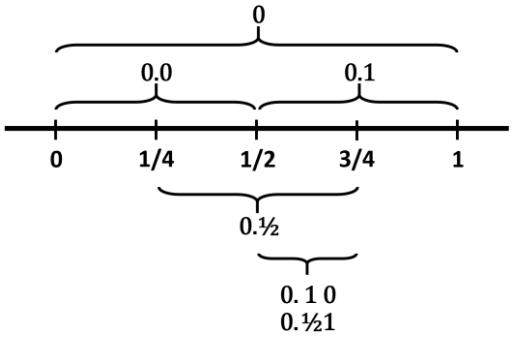

Fig. 3. Intervals corresponding to digits in systems with base 2 and additional digit ½.

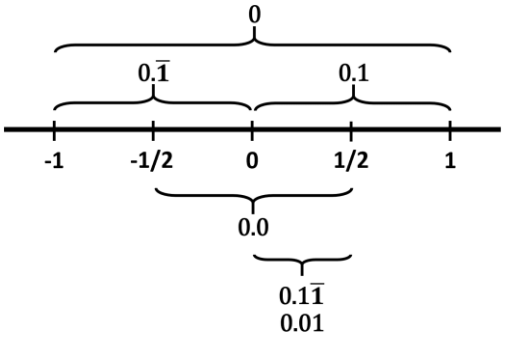

Fig. 4. Intervals corresponding to digits in symmetric system with base 2 and three digits $\overline{1}$, 0, 1.

As is known from the theory and practice of conventional positional systems, symmetric systems with opposite numbers (like the one used by Cauchy [3]) have certain advantages. First, the negation of a number is trivial, and the subtraction algorithm is completely analogous and symmetric to addition. Secondly, the rounding error, in our case, the intervals, is calculated symmetrically, without bias in any direction. Symmetry requires an odd number of digits and is easily implemented, for example, in the ternary numeration system with digits –1, 0, 1, as was the case in the Soviet computer "Setun" [2]. It is also beneficial to do this in a binary system with overlaying based on three digits.

Figure 4 shows a system with numbers $\overline{1}$, 0, 1. Compared with Figure 3 all labels and designations are shifted by –½ and multiplied by 2. As in Figure 3, the interval indicated by the lower parenthesis in Figure 4 is written in two ways: $0.1\overline{1}$ and 0.01.

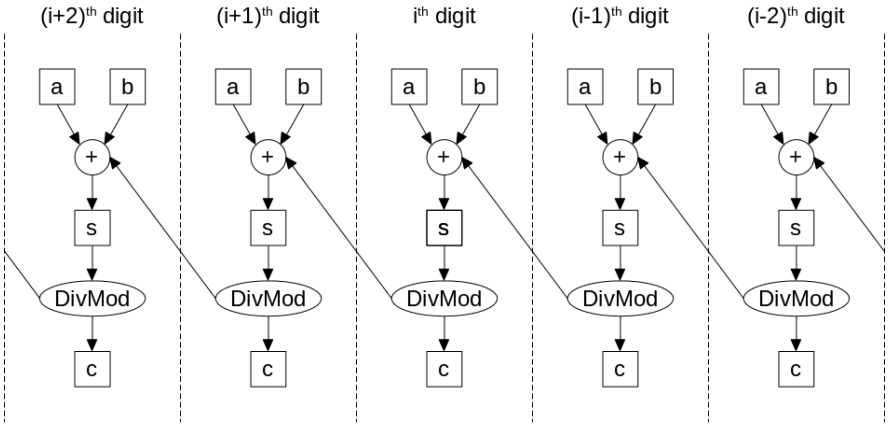

Fig. 5. Addition in the classical positional numeration system.

3 Addition Schemes

Let us illustrate how addition schemes with carry look in classical numeration systems and in overlaying systems with extra digits.

Figure 5 shows the simplest classical scheme for the bitwise addition of two numbers. Inputs *a* and *b* are the corresponding digits of two numbers. On the left are the most significant digits, on the right are the least significant ones. The digits are numbered in ascending order towards the least significant digits. Since the sum of *s* digits *a* and *b* can be greater than the base of the numeration system, it is split into a hyphen and a digit by the operation DivMod modulo the base of the numeration system. The diagram clearly shows the path of potentially unlimited carry from right to left: ... +, s, DivMod, +, s, DivMod, +, s, DivMod ...

Figure 6 shows the computational schemes for the symmetrical “three-halves” overlay addition system from Figure 4. Here the DivMod operation is replaced by two operations – carry Π and the weighted sum Σ, calculated by the following formulas:

$$\Pi(x, y) = (2x + y + 1)/4,$$
$$\Sigma(x, y, z) = -2x + y + z.$$

The carry is now limited to two digits. All calculation paths are finite and have the form: +, s, Π, Σ.

The diagram can be simplified by using a base of at least 3 and two additional digits. It is shown in Figure 7. Here the carry is "absorbed" by one digit position. This happens when the system has a base of at least 3 and contains two additional digits. Here the calculation paths are +, s, DivMod, +.

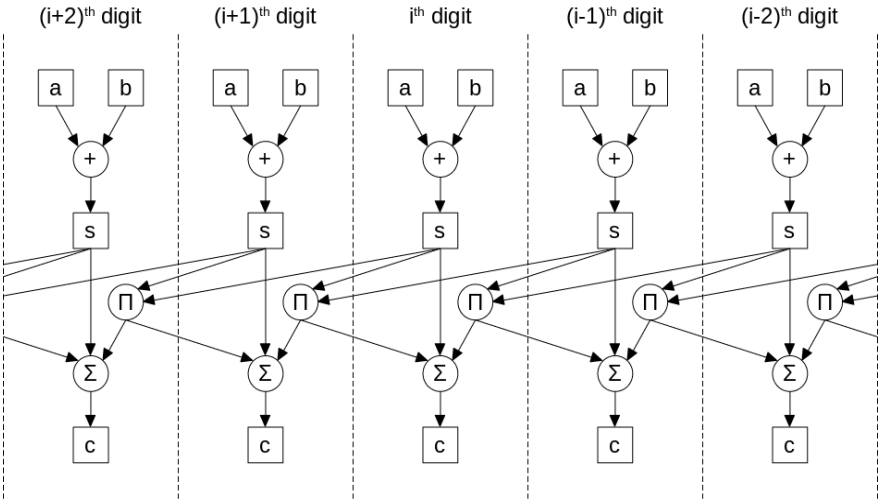

Fig. 6. Overlaid positional addition with one additional digit.

4 Conclusions

The paper and the report provide an introduction to the topic of high-precision arithmetic with redundant representation of numbers and the concept of an overlaid numeration system, in which digit-wise parallelism of addition and subtraction of real numbers is possible due to limited carry from a digit position to the next ones. Multiplication and division are also speeded up as they use parallel addition and subtraction.

Computational schemes were presented with a carry by 2 digits for the "three halves" system with one additional digit compared to binary and with a carry by one position for systems with two additional digits and with a base of at least 3.

We are developing circuit designed solutions for the implementation of such arithmetic operations in FPGAs with the further aim of conducting an experimental study of the effect and usefulness of such arithmetic on various applied problems.

It is surprising that despite a rather long history of theoretical study of such systems, we do not know that they have been brought to a convenient practical implementation and widespread testing on applied problems. Perhaps, with all attempts, the desired efficiency was not achieved, and negative results were obtained, which, unfortunately, are not accepted to be published in modern science. On the other hand, it is possible that such arithmetic was inadequate to the previous level of hardware development. For example, today's large volume of circuits loaded into FPGAs creates qualitatively new opportunities than were available before. Understanding the algorithmic problems that still need to be solved and realizing the usefulness only for a limited class of problems, we nevertheless believe that it is necessary to provide users with an implementation of

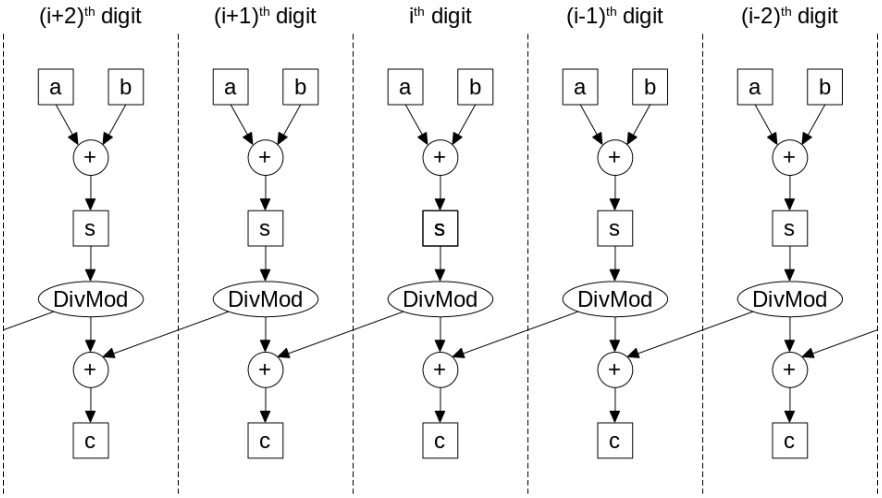

Fig. 7. Addition in overlaid positional system with two redundant digits.

long numbers based on overlaying systems (as well as other representations) in a way convenient for programmers for easy experimentation with problems that require high precision and interval arithmetic.

For applied mathematicians, the ability to freely experiment with new tools is a key factor in their development and bringing them to widespread practice. Mathematicians do not hesitate to answer "yes, come on!" when asked if you want to recalculate your problem with high accuracy and find out how much rounding errors affect the result, even if you need to leave it on a (super) computer overnight instead of getting an answer within an hour on a laptop. But as soon as you tell them that for this you need to rewrite their favorite program in C / C ++ or Fortran, replacing operations in arithmetic expressions with calls to library functions, or recode the program in another language like Python, in which long arithmetic is immediately built into realization, another reaction follows: “no, don't; I'd better go and think."

New high-precision arithmetic – be it modular, overlapped, or simply long-represented in a positional system, whether implemented in a central processing unit or in an accelerator like GPGPU or FPGA - must be seamlessly integrated with popular languages. The transition from programs with ordinary arithmetic (on the types int, long, float and double) to programs with high-precision arithmetic should, in the simplest case, be carried out only by setting some options to the compiler. The implementation of such tools is not a task of developing new representations, algorithms and programs for operations on numbers (which this paper is devoted to), but a common system programming task that requires known resources. Unfortunately, we are not aware of such systems. Their absence is one of the possible obstacles to the development of this direction.

In addition to various mathematical problems, where long arithmetic in hundreds, thousands or more bits can help to obtain new information from a computational experiment, we will point out two more classes where the overlaying numeration system can be advantageous:

1. Overlaying arithmetic is essentially interval arithmetic: a finite set of digits has the semantics of an interval to which all numbers beginning with this set belong. Thus, it is applicable wherever interval arithmetic is required to assess the accuracy of calculations. Such arithmetic has a fundamental drawback: the intervals "spread out" too quickly and often do not provide any useful information. Therefore, interval arithmetic should be used in conjunction with long numbers so that it can be easily recalculated with greater precision, when the accuracy of the current result is insufficient.
2. Operations in overlaying systems are performed from the most significant bits to the least significant ones. (Recall that it was for this reason that L.E. Brouwer used it [1]). Due to this, arithmetic expressions can be implemented by pipelined circuits for FPGAs, "pumping" long numbers from high-order digits to low-order ones. It will be parallelism of a different type: not the simultaneous processing of many digits, which was discussed above, but sequential, pipelined. In this case, it becomes possible to control the accuracy dynamically, "pulling" from the computational schema as many result digits as needed, "pumping up" the required number of argument digits.

These are topics for future research.

5 Acknowledgements

The authors express their gratitude to Arkady Klimov for valuable advice on improving the presentation and paper.

This work was financially supported by the Russian Federation represented by Ministry of Education and Science (grant id: RFMEFI61319X0092).

References

1. Brouwer, L.E.J.: Besitzt jede reelle Zahl eine Dezimalbruchentwicklung? (Does Every Real Number Have a Decimal Expansion?). Mathematische Annalen, 83(3–4), 201–210 (1921). DOI: 10.1007/BF01458382. English translation in: Mancosu P. (ed.), From Brouwer to Hilbert. The Debate on the Foundations of Mathematics in the 1920s. Oxford: Oxford University Press, P. 28–35 (1998).
2. Brusentsov, N.P., Alvarez, J.R.: Ternary Computers: The Setun and the Setun 70 Perspectives on Soviet and Russian Computing, SoRuCom-2006, IFIP Advances in Information and Communication Technology, Vol 357, Berlin, Heidelberg: Springer, 2011, DOI: 10.1007/978-3-642-22816-2_10.
3. Cauchy, A.-L.: Sur les moyens d'éviter les erreurs dans les calculs numériques (On Ways to Avoid Errors in Numerical Calculations), Comptes rendus de l'Académie des Sciences, XI,

789–798 (1840). Reprinted in: Cauchy A.-L. Œuvres complètes, série 1, 5, 431–442 (1885). URL: http://sites.mathdoc.fr/cgi-bin/oeitem?id=OE_CAUCHY_1_5_431_1.
4. Frougny, Ch., Sakarovitch, J.: Number Representation and Finite Automata. Berthé V., Rigo M. (Eds.), Combinatorics, Automata and Number Theory (Encyclopedia of Mathematics and its Applications, 135), Cambridge: Cambridge University Press, 34–107 (2010). DOI: 10.1017/CBO9780511777653.003.
5. Frougny, Ch., Pelantová, E., Svobodová, M.: Parallel Addition in Non-Standard Numeration Systems. Theoretical Computer Science, 412(41), 5714–5727 (2011), DOI: 10.1016/j.tcs.2011.06.028.
6. Isupov, K.S., Knyazkov, V.S.: Parallel multiple-precision arithmetic based on residue number system. Program Systems: Theory and Applications, 7:1(28), 61–97 (2016), DOI: 10.25209/2079-3316-2016-7-1-61-97.
7. Isupov, K.S., Knyazkov, V.S.: Multiple-precision matrix-vector multiplication on graphics processing units. Program Systems: Theory and Applications, 11:3(46), 33–59 (2020), DOI: 10.25209/2079-3316-2020-11-3-33-59.
8. Cormen, Th.H., Leiserson, Ch.E., Rivest, R.L.: Introduction to Algorithms, Cambridge, MA: MIT Press, 1990, 1048 p. Russian translation: M.: MCNMO, 1999, 960 p. ISBN 5-900916-37-5. URL: https://istina.msu.ru/publications/book/171753101/.
9. Malashevich, B.M.: Brief fundamentals and history of the creation of domestic modular computers. The origins of modular arithmetic. Proceedings of the SoRuCom-2a017. Forth International Conference “Computer Technology in Russia and in the Former Soviet Union”, Zelenograd, October 3–5. M., 2017, p. 193–207, URL: https://www.computer-museum.ru/books/SORUCOM_2017_1.pdf.
10. Nepejvoda, N.N.: Constructive mathematics: overview of advantages, disadvantages and lessons I, Logical Investigations: Almanac, 17, 191–239 (2011), DOI: 10.21146/2074-1472-2011-17-0-191-239.
11. Nepejvoda, N.N.: Additive representations of numbers: some remarks. Program Systems: Theory and Applications, 8:4(35), 101–115 (2017), DOI: 10.25209/2079-3316-2017-8-4-101-115.
12. Nepejvoda, N.N., Grigorevsky, I.N., Lilitko, E.P.: On representation of real numbers. Program Systems: Theory and Applications, 5:4(22), 105–121 (2014). URL: http://psta.psiras.ru/read/psta2014_4_105-121.pdf.
13. Nepejvoda, N.N., Grigorevsky, I.N.: Some Issues on Linear Numeration Systems. IOSR Journal of Mathematics (IOSR-JM), 15(6), Series 4, 47–50 (2019), URL: http://www.iosrjournals.org/iosr-jm/papers/Vol15-issue6/Series-4/E1506044750.pdf.
14. Nepejvoda, N.N., Grigorevsky, I.N., Klimov, And.V., Klimov, Yu.A., Romanenko, S.A.: Computational Aspects of Various Number Representations. Test Engineering and Management, 83, 28224–28231 (2020).
URL: http://www.testmagzine.biz/index.php/testmagzine/article/view/12901.
15. Shvorin, A.B.: Parallel Addition of Real Numbers in Overlaying Numeration Systems. Program Systems: Theory and Applications, 6:2(25), 101–117 (2015). DOI: 10.25209/2079-3316-2015-6-2-101-117.
16. Shvorin, A.B.: Digit-Wise Parallel Addition of Real Numbers in Overlaying Numeration Systems. Asian Academic Research Journal, 3(1) (2016),
URL: https://pat.keldysh.ru/~art/publications/2016-01-Shvorin-Parallel-Addition.pdf.