ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

UDC 519.857:517.956 **doi: 10.32620/reks.2021.4.03**

V. MAKARICHEV, V. KHARCHENKO

National Aerospace University "Kharkiv Aviation Institute", Ukraine

APPLICATION OF DYNAMIC PROGRAMMING APPROACH TO COMPUTATION OF ATOMIC FUNCTIONS

The special class of atomic functions is considered. The atomic function is a solution with compact support of linear differential functional equation with constant coefficients and linear transformations of the argument. The functions considered are used in discrete atomic compression (DAC) of digital images. The algorithm DAC is lossy and provides better compression than JPEG, which is de facto a standard for compression of digital photos, with the same quality of the result. Application of high precision values of atomic functions can improve the efficiency of DAC, as well as provide the development of new technologies for data processing and analysis. This paper aims to develop a low complexity algorithm for computing precise values of the atomic functions considered. Precise values of atomic functions at the point of dense grids are the ***subject matter*** *of this paper. Formulas of V. O. Rvachev and their generalizations are used. Direct application of them to the computation of atomic functions on dense grids leads to multiple calculations of a great number of similar expressions that should be reduced. In this research, the reduction required is provided. The* ***goal*** *is to develop an algorithm based on V. O. Rvachev's formulas and their generalizations. The following* ***tasks*** *are solved: to convert these formulas to reduce the number of arithmetic operations and to develop a verification procedure that can be used to check results. In the current research,* ***methods*** *of atomic function theory and dynamic programming algorithms development principles are applied. A numerical scheme for computation of atomic functions at the points of the grid with the step, which is less than each predetermined positive real number, is obtained and a dynamic algorithm based on it is developed. Also, a verification procedure, which is based on the properties of atomic functions, is introduced. The following* ***results*** *are obtained: 1) the algorithm developed provides faster computation than direct application of the corresponding formulas; 2) the algorithm proposed provides precise computation of atomic functions values; 3) procedure of verification has linear complexity in the number of values to be checked. Moreover, the algorithms proposed are implemented using Python programming language and a set of tables of atomic functions values are obtained.* ***Conclusions:*** *results of this research are expected to improve existing data processing technologies based on atomic functions, especially the algorithm DAC, and accelerate the development of new ones.*

Keywords: *atomic functions; up-function; dynamic programming; verification; discrete atomic compression.*

Introduction

The rapid development of computational technologies provides the possibility to solve different tasks of such branches, as data processing and artificial intelligence. At the same time, it has led to a set of new problems concerning big data, as well as challenges due to cybercrime. Therefore, the development and implementation of new technologies for solving these problems are relevant. For this purpose, different approaches, in particular, the application of non-classic mathematical tools can be used.

In the 1970-s, V. O. Rvachev and V. L. Rvachev introduced atomic function theory [1]. Various features of these functions including approximation properties were studied in [2 – 4]. A function is called atomic if it is a compactly supported solution of linear functional differential equation with constant coefficients and linear transformations of the argument [4]. Atomic functions were designed to eliminate limitations of such classic constructive tools, as trigonometric and algebraic polynomials, as well as splines. At that time, the theory of atomic functions was developed taking into account the rather low computational capabilities. A set of fundamental results, which were obtained by V. O. Rvachev and representatives of his scientific school, provided the successful application of atomic functions to different processes and phenomena modeling [5, 6], as well as lossy compression of digital images [7, 8]. Also, it was shown in [9, 10] that lossless and near-lossless image compression can be obtained. A combination of convenient properties makes it promising to apply atomic functions to the development of high-performance data analysis and processing algorithms.

This paper is devoted to atomic functions computation issue. We consider the functions

$$\mathrm{up}_s(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{itx}\prod_{k=1}^{\infty}\frac{\sin^2\left(\frac{st}{(2s)^k}\right)}{\frac{t}{(2s)^k}s^2\sin\left(\frac{t}{(2s)^k}\right)}dt,$$

where s = 1, 2, 3, ….

For the case $s=1$, $up_s(x)$ is well-known up-function of V. O. Rvachev [4]. For the case $s \geq 2$, these functions were presented by V. O. Rvachev and G. O. Starets [11]. In [7, 8], V. V. Lukin, I. V. Brysina and V. O. Makarichev introduced and investigated efficiency of discrete atomic compression (DAC) that is image compression algorithm based on application of atomic functions $up_s(x)$. A combination of such features of these functions as smoothness, finiteness and good approximation properties [4, 12] provides an advantage of the algorithm DAC over analogues, in particular, JPEG that is de facto a standard for compression of digital photos.

In the current research, we solve the problem of $up_s(x)$ computation and verification of the results. **The aim of this paper** is to develop an algorithm, which has low complexity and provides computation of $up_s(x)$ at the points of grid with a step that is less than any predetermined small number. In future, it will improve the algorithm DAC and make it easier to apply atomic functions $up_s(x)$.

The paper structure is the following. Section 1 discusses existing methods of $up_s(x)$ computation. Sections 2 and 3 describe atomic function computation algorithm based on dynamic programming principle and rules of the verification. Numerical experiments are discusses in sections 4 and 5. The last section concludes research and describes future steps.

1. Formulation of the problem

Consider the case $s=1$. We get the function

$$up(x)=\frac{1}{2\pi}\int_{-\infty}^{\infty} e^{itx}\prod_{k=1}^{\infty}\frac{\sin\left(t2^{-k}\right)}{t2^{-k}}dt,$$

which is a solution with a compact support of the equation

$$y'(x)=2y(2x+1)-2y(2x-1).$$

Graph of this function and its derivative are shown in Fig.1.

Development of atomic functions theory started in 1971 from the research of up(x). Further, other atomic functions were constructed and investigated.

To compute values of up-function, V. O. Rvachev obtained the following series [3]:

$$up(x-1)=\sum_{k=1}^{\infty}(-1)^{s_k+1}p_k\times$$

$$\times\sum_{j=0}^{k}\frac{2^{j(j+1)/2}b_{k-j-1}}{2^{(k-j-1)(k-j)/2}(k-j-1)!\,j!}(x-0,p_1...p_k)^j, \quad (1)$$

where $(0,p_1...p_k)$ is binary form of x and

$$s_k=\sum_{j=1}^{k}p_j,\ b_{-1}=1,\ b_k=\int_0^1 x^k up(x)dx.$$

Fast convergence of the series (1) is provided by the following estimate of the remainder $R_n(x)$:

$$|R_n(x)|\leq\frac{C}{2^{n(n-1)/2+2[(n-1)/2](2[(n-1)/2]+1)}(n-1)!},$$

where C is some positive constant and [z] is an integer part of the number z.

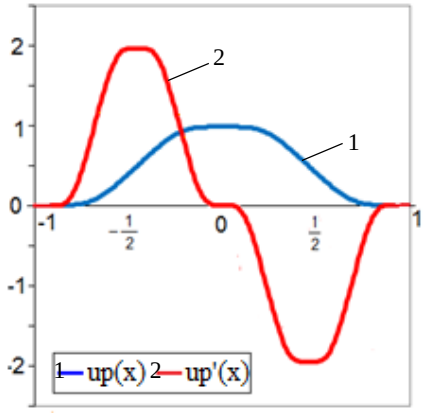

Fig. 1. Graphs of the function up(x) and its derivative: similarity of the function and its derivative can be seen

Using (1), values of atomic function up(x) can be calculated with any predetermined accuracy. Nevertheless, computational errors tend to accumulate and impair results. Hence, possibility to minimize errors is desired. For instance, in lossy image compression algorithms, quantization procedure and arithmetic errors, which are obtained due to application of some discrete data transform, provide loss of quality. Efficiency of such algorithms can be improved by usage of more accurate values of mathematical functions applied. So, it is reasonable to consider another approach to computation of atomic functions.

In [3], V. A. Rvachev obtained the following formulas:

$$up\left(-1+\frac{k}{2^n}\right)=\frac{1}{n!2^{n(n-1)/2}}\sum_{j=1}^{k}\delta_j\times$$

$$\times\sum_{i=0}^{[n/2]}\binom{n}{2i}\left(k-j+\frac{1}{2}\right)^{n-2i}\mu_{2i}2^{-2i}, \quad (2)$$

where n is positive integer, $k=0,1,...,2^{n+1}$, coefficients $\{\delta_s\}$ satisfies recursive formulas:

$$\delta_1=1,\ \delta_{2i-1}=\delta_i,\ \delta_{2i+1}=-\delta_i \text{ for } k=1,2,...,$$

and, finally,

$$\mu_{2i}=\int_{-1}^{1}x^{2i}up(x)dx.$$

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Moments μ_{2i} of the function up(x) satisfy

$$\mu_0 = 1,\ \mu_{2i} = \frac{1}{2^{2i}-1}\sum_{k=1}^{i}\binom{2i}{2k}\frac{\mu_{2i-2k}}{2k+1} \qquad (3)$$

for $i = 1, 2, \ldots$.

It follows from (2) in combination with (3) that up(x) can be computed precisely at the points of grid with a step, which is less than any predetermined small number (for this purpose, n should be chosen large enough).

Formulas (2), (3) were generalized by G. Starets for atomic functions $up_s(x)$, $s = 2, 3, 4, \ldots$.

Let $x_{s,n,j} = -1 + \frac{j}{s(2s)^n}$, $j = 0, 1, \ldots, (2s)^{n+1}$. In [13] and further in [14], the following expressions were obtained:

$$up_s(x_{s,n,j}) = \frac{2^{n+1}}{n!(2s)^{(n+1)(n+2)/2}}\sum_{p=1}^{j}\delta_{s,p}\times$$

$$\times\sum_{k=0}^{[n/2]}\binom{n}{2k}(2j-2p+1)^{n-2k}\mu_{2k}, \qquad (4)$$

where $\mu_{s,0} = 1$ and for any positive integer p the following recursive formulas hold:

$$\mu_{s,2p} = \frac{1}{s^2\left((2s)^{2p}-1\right)}\sum_{k=1}^{p}\binom{2p}{2k}\frac{\mu_{s,2p-2k}}{2k+1}\times$$

$$\times\sum_{i=1}^{s}(2i-1)^{2k+1}. \qquad (5)$$

Also, coefficients $\{\delta_{s,j}\}$ can be found recursively:

$$\delta_{s,i} = 1,\ \delta_{s,s+i} = -1 \text{ for } i = 1, \ldots, s, \qquad (6)$$

$$\delta_{s,2s(p-1)+j} = \delta_{s,p}\cdot\delta_{s,j} \qquad (7)$$

for each $i = 2, 3, \ldots, n$, $p = 2, 3, \ldots, (2s)^{i-1}$ and $j = 1, \ldots, 2s$.

Some reduction of computation complexity is provided by properties of $up_s(x)$ [13 – 15]. This implies that each of these functions is even. Therefore,

$$up_s\left(x_{s,n,j}\right) = up_s\left(-x_{s,n,j}\right) =$$

$$= up_s\left(-1+\frac{(2s)^{n+1}-j}{s(2s)^n}\right) = up_s\left(x_{s,n,(2s)^{n+1}-j}\right). \qquad (8)$$

Furthermore,

$$up_s\left(x_{s,n,k(2s)^n+j}\right) = \frac{k}{s} + up_s\left(x_{s,n,j}\right) \qquad (9)$$

for any $k = 0, 1, \ldots, s-1$ and $j = 0, 1, \ldots, (2s)^n$.

Whence, to obtain values of the function $up_s(x)$ at the points $x_{s,n,j}$ for any $j = 0, 1, \ldots, (2s)^{n+1}$, it is sufficient to apply (4) – (7) just for $j = 0, 1, \ldots, (2s)^n$; and then values of $up_s(x)$ can be found at all other points $x_{s,n,j}$ using (8), (9),

Denote by $X_{s,n}$ the set of points

$$\left\{x_{s,n,j} : j = 0, 1, \ldots, (2s)^n\right\}.$$

It can be easily shown that if we apply formulas (4) – (7) directly for any point x of the set $X_{s,n}$, we get an algorithm with lower complexity estimate of

$$O\left(\left|X_{s,n}\right|^2 \log_{2s}^2\left|X_{s,n}\right|\right), \qquad (10)$$

where $\left|X_{s,n}\right|$ is number of elements of this set. Besides, calculation of similar or even identical expressions is present. Hence, the direct application considered is unreasonable.

To get complexity reduction, dynamic programming approach can be applied. Its main idea can be described as follows [16]: if process of some problem solving contains many identical subproblems, then complexity can be reduced by storing their solutions in memory to avoid solving them multiple times. It is this approach that can be applied to development of up_s-values computation algorithm with complexity, which is less than (10).

The main task of this paper is to develop algorithm for computation of values of the function $up_s(x)$ on the set $X_{s,n}$, using dynamic programming principles. We construct verification rules and illustrate results of checking as well.

2. Dynamic algorithm for computation of atomic functions $up_s(x)$

In this subsection, we develop atomic functions $up_s(x)$ computation algorithm based on dynamic programming approach. For this purpose, we suggest application of formulas (4) – (7) in combination with some modifications.

Let s be fixed positive integer. Also, we consider the case n = 2m, where m = 1, 2, … . We stress that the following approach also can be extended for the case n = 2m – 1.

First, it follows from (4) that computation of $up_s(x_{s,2m,j})$ requires values $\{\delta_{s,p}\}$ and $\{\mu_{s,2k}\}$. Hence, these values should be calculated before computation of up_s-values and stored in arrays. Moreover, in both formulas (4) and (5), a set of binomial coefficients $\binom{k}{j}$ is used. So, it is reasonable to obtain them before the whole computation process in order to reduce computation complexity. To calculate binomial coefficients re-

quired, one can apply dynamic programming algorithm, which is based on the following well-known formulas:

$$\binom{k}{0}=\binom{k}{k},\ \binom{k}{j}=\binom{k-1}{j-1}+\binom{k-1}{j}. \qquad (11)$$

Second, consider recursive formula (5). Here, we see that sums $S_\mu(k)=\sum_{i=1}^{s}(2i-1)^{2k+1}$ are used. It is suggested to compute them and after that to obtain values $\mu_{s,2k}$.

Now, we modify formula (4) in order to decrease a number of operations.

If we put n = 2m, we get

$$up_s\left(x_{s,2m,j}\right)=\frac{2^{2m+1}}{(2m)!(2s)^{(2m+1)(m+1)}}\sum_{p=1}^{j}\delta_{s,p}\times$$

$$\times\sum_{k=0}^{m}\binom{2m}{2k}(2j-2p+1)^{2(m-k)}\mu_{s,2k}\ .$$

If we apply the change of index q = m – k, we obtain

$$up_s\left(x_{s,2m,j}\right)=\frac{2^{2m+1}}{(2m)!(2s)^{(2m+1)(m+1)}}\sum_{p=1}^{j}\delta_{s,p}\times$$

$$\times\sum_{q=0}^{m}\binom{2m}{2k}(2j-2p+1)^{2q}\mu_{s,2m-2q}\ .$$

Here, we note that the identity

$$\binom{2m}{2m-2q}=\binom{2m}{2q}$$

is applied as well.

This implies that

$$up_s\left(x_{s,2m,j}\right)=\frac{2^{2m+1}}{(2m)!(2s)^{(2m+1)(m+1)}}\times$$

$$\times\sum_{q=0}^{m}\binom{2m}{2q}\cdot S_{up}(2q,j)\cdot\mu_{s,2m-2q}\ , \qquad (12)$$

where

$$S_{up}(r,j)=\sum_{p=1}^{j}(2j-2p+1)^r\cdot\delta_{s,p}\ .$$

Further, for the case j > 1 we get

$$S_{up}(r,j)=\sum_{p=1}^{j-1}(2j-2p+1)^r\cdot\delta_{s,p}+\delta_{s,j}=$$

$$=\sum_{p=1}^{j-1}\left((2(j-1)-2p+1)+2\right)^r\cdot\delta_{s,p}+\delta_{s,j}=$$

$$=\sum_{p=1}^{j-1}\delta_{s,p}\sum_{i=0}^{r}\binom{r}{i}2^{r-i}(2(j-1)-2p+1)^i+\delta_{s,j}=$$

$$=\sum_{i=0}^{r}\binom{r}{i}2^{r-i}\sum_{p=1}^{j-1}\delta_{s,p}(2(j-1)-2p+1)^i+\delta_{s,j}=$$

$$=\sum_{i=0}^{r}\binom{r}{i}2^{r-i}S_{up}(i,j-1)+\delta_{s,j}\ .$$

So,

$$S_{up}(r,j)=\sum_{i=0}^{r}\binom{r}{i}2^{r-i}S_{up}(i,j-1)+\delta_{s,j} \qquad (13)$$

for any r = 0, 1, …, 2m and j = 2, 3, …, $(2s)^{2m}$.

Since $\delta_{s,1}=1$ (see (6)), we get

$$S_{up}(r,1)=1 \qquad (14)$$

for each r = 0, 1, …, 2m.

Whence, a numerical scheme for computation of $up_s(x_{s,2m,j})$ is

1) compute and store binomial coefficients $\binom{a}{b}$ ((11) can be applied for this purpose);

2) compute moments $\mu_{s,2k}$ of the function $up_s(x)$:

2.1) calculate sums $S_\mu(k)$;

2.2) calculate $\mu_{s,2k}$, using (5);

3) compute coefficients $\delta_{s,j}$, using (6), (7);

4) compute values $up_s(x_{s,2m,j})$:

4.1) calculate sums $S_{up}(r, j)$, using (13), (14);

4.2) calculate $up_s(x_{s,2m,j})$, using (12).

In more detail, **dynamic algorithm** for computation of values of the function $up_s(x)$ on the set $X_{s,2m}$ can be represented as follows:

1) compute $\binom{a}{b}$ for any a = 0, 1, …, 2m and b = 0, 1, …, a:

1.1) for each a = 0, 1, …, 2m

$$\binom{a}{0}=\binom{a}{a}=1\ ;$$

1.2) for any b = 1, 2, …, 2m and a = b+1,…, 2m

$$\binom{a}{b}=\binom{a-1}{b-1}+\binom{a-1}{b}\ ;$$

2) compute $\mu_{s,2p}$ for p = 0, 1, …, m:

2.1) for each k = 1, 2, …, m calculate

$$S_\mu(k)=\sum_{i=1}^{s}(2i-1)^{2k+1}\ ;$$

2.2) $\mu_{s,0}=1$;

2.3) for any p = 1, 2, …, m

$$\mu_{s,2p}=\frac{1}{s^2\left((2s)^{2p}-1\right)}\sum_{k=1}^{p}\binom{2p}{2k}\cdot\frac{\mu_{s,2p-2k}}{2k+1}\cdot S_\mu(k)\ ;$$

3) calculate $\delta_{s,j}$ for any j = 0, 1, …, $(2s)^{2m}$:

3.1) for each i = 1, …, s

$$\delta_{s,i}=1,\ \delta_{s,s+i}=-1\ ;$$

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

3.2) for any i = 2, 3, …, 2m, p = 2, 3, …, $(2s)^{i-1}$ and j = 1, 2, …, 2s

$$\delta_{s,2s(p-1)+j} = \delta_{s,p} \cdot \delta_{s,j};$$

4) for each j = 0, 1, …, $(2s)^{2m}$ compute values $up_s(x_{s,2m,j})$:

4.1) for any r = 0, 1,…,2m and j = 0, 1,…, $(2s)^{2m}$ compute $S_{up}(r, j)$:

4.1.1) $S_{up}(r, 1) = 1$ for each r = 0, 1, …, 2m;

4.1.2) for any j = 2, 3,…, $(2s)^{2m}$ and any r = 0, 1,…, 2m

$$S_{up}(r, j) = \sum_{i=0}^{r} \binom{r}{i} 2^{r-i} S_{up}(i, j-1) + \delta_{s,j};$$

4.2) $up_s(x_{s,2m,0}) = 0$;

4.3) compute the constant

$$C_{s,m} = \frac{2^{2m+1}}{(2m)!(2s)^{(2m+1)(m+1)}};$$

4.4) for each j = 1, 2, …, $(2s)^{2m}$

$$up_s\left(x_{s,2m,j}\right) = C_{s,m} \sum_{q=0}^{m} \binom{2m}{2q} \cdot S_{up}(2q, j) \cdot \mu_{s,2m-2q}.$$

The proposed algorithm provides computation of values of the function $up_s(x)$ at the points of grid with the step $h_{s,m} = \frac{1}{s(2s)^{2m}}$. It is obvious that for any fixed s this step quickly converges to 0 as $m \to \infty$. This means that by fixing an appropriate value of m one can get $h_{s,m}$, which is less than any predetermined positive real number.

In other words, values of atomic function $up_s(x)$ can be obtained on an arbitrarily dense grid. Moreover, these values can be computed precisely.

Finally, it follows that complexity T(s, m) of the proposed algorithm can be expressed as follows:

$$T(s, m) = O\left(m^2 (2s)^{2m}\right).$$

Since $\left|X_{s,2m}\right| = (2s)^{2m} + 1$, we get

$$T(s, m) = O\left(\left|X_{s,2m}\right| \cdot \log_{2s}^2 \left|X_{s,2m}\right|\right). \quad (15)$$

It is clear that for any s the function T(s, m) grows exponentially. However, the step $h_{s,m}$ of the grid $X_{s,2m}$ decreases exponentially. This means that if m is too large, then $X_{s,2m}$ is too thick, which is rarely required.

Comparing (10) with (15), we conclude that the algorithm, which is developed above, has much less time complexity. In other words, the proposed algorithm provides faster computation of up_s-function values. Nevertheless, it is clear that such acceleration requires additional memory expenses, especially, for storing of sums $S_{up}(r, j)$.

This feature is common for algorithms that are developed using dynamic programming principles.

3. Verification of the algorithm

The algorithm proposed in the previous subsection provides computation of atomic function $up_s(x)$ at the points of the fixed grid. Its output is an array of values of this function. Numerous features of $up_s(x)$ can be applied to verify the results obtained. Here, we consider the most appropriate one. An approach to verification is based on specifying algorithm invariants similar to [19, 20].

V. O. Rvachev proved the following [3]:

$$\sum_{k=-\infty}^{\infty} up(x-k) \equiv 1.$$

Further, this identity was generalized in [11]:

$$\sum_{k=-\infty}^{\infty} up_s(x-k) \equiv 1, \; s = 1, 2, 3, \ldots. \quad (16)$$

In other words, sum of shifts of up_s-function with the step 1 equals 1 (see Figures 2 and 3).

Now, we apply (16) to obtain verification rule.

Since $up_s(x) = 0$ for any $x \notin (-1, 1)$, we get

$$up_s(x) + up_s(x+1) \equiv 1 \quad (17)$$

for any $x \in [-1, 0]$.

Consider the point $x_{s,2m,j} = -1 + \frac{j}{s(2s)^{2m}}$, where $j = 0, 1, \ldots, (2s)^{2m}$. We get

$$up_s\left(x_{s,2n,j} + 1\right) = up_s\left(\frac{j}{s(2s)^{2m}}\right) = \left|\begin{matrix} up_s(x) \text{ is} \\ \text{even} \end{matrix}\right| =$$

$$= up_s\left(-\frac{j}{s(2s)^{2m}}\right) = up_s\left(-1 + \frac{s(2s)^{2m} - j}{s(2s)^{2m}}\right) =$$

$$= up_s\left(-1 + \frac{(s-1)(2s)^{2m} + (2s)^{2m} - j}{s(2s)^{2m}}\right) =$$

$$= up_s\left(-1 + \frac{s-1}{s} + \frac{(2s)^{2m} - j}{s(2s)^{2m}}\right) \overset{(9)}{=} \frac{s-1}{s} +$$

$$+ up_s\left(-1 + \frac{(2s)^{2m} - j}{s(2s)^{2m}}\right) = \frac{s-1}{s} + up_s\left(x_{s,2m,(2s)^{2m}-j}\right).$$

Hence,

$$up_s\left(x_{s,2n,j} + 1\right) = \frac{s-1}{s} + up_s\left(x_{s,2m,(2s)^{2m}-j}\right).$$

Combining this with (17), we obtain

$$up_s\left(x_{s,2m,j}\right) + \frac{s-1}{s} + up_s\left(x_{s,2m,(2s)^{2m}-j}\right) = 1$$

that provides the following:

$$\left(up_s\left(x_{s,2m,j}\right) + up_s\left(x_{s,2m,(2s)^{2m}-j}\right)\right) \cdot s = 1 \quad (18)$$

for any $j = 0, 1, \ldots, (2s)^{2m}$.

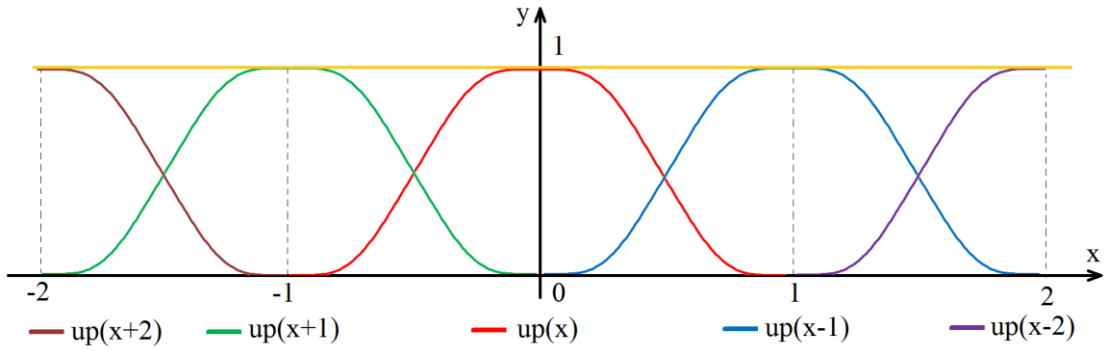

Fig. 2. Graphs of up(x) and its shifts

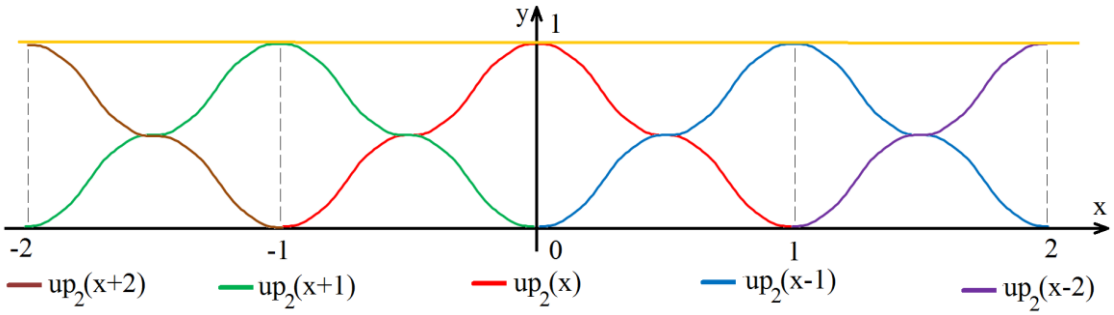

Fig. 3. Graphs of $up_2(x)$ and its shifts

To verify all values $up_s\left(x_{s,2m,j}\right)$, we suggest to apply maximum absolute deviation (MAD):

$$MAD = \max_{j=0,1,\ldots,(2s)^{2m}} \left| \left(up_s\left(x_{s,2m,j}\right) + up_s\left(x_{s,2m,(2s)^{2m}-j}\right) \right) \cdot s - 1 \right|.$$

Notice that this metric is very sensitive to any minor errors. If it is equal or close to 0, then validation can be considered successful.

Complexity of the MAD-metric computation is linear over the number of values to be checked. Moreover, this procedure can be reduced, since each term except $j = (2s)^{2m}/2$ is considered twice. So, maximum should be evaluated over $j = 0,1,\ldots,(2s)^{2m}/2$.

4. Numerical experiments

In the current research, we've implement the algorithm, which was proposed above, using Python 3.9 (https://www.python.org). It provides long arithmetic, contains rational fractions manipulation tools and is freely usable. We've applied built-in type float for floating point values of $up_s(x)$, as well as Fraction from the fractions module for precise values of these functions.

Using the software developed, we've obtained sets of values of $up_s(x)$ on $X_{s,2m}$ for parameters that are presented in Table 1. Files with floating point and fractional values are available at the link to Google Drive: https://drive.google.com/drive/folders/1mqM4uKRJKG5cjPRgbRfX_hZSAOluKvQY?usp=sharing.

Also, we've compared floating point values of $up_s(x_{s,2m,j})$ with the corresponding fractional ones. In Table 2, results of comparison are shown. Here, we note that maximum absolute deviation of floating point values from fractional ones is applied as a metric of difference.

Table 1

Parameters of the values obtained

s	m	step of the grid $X_{s,2m}$
1	8	1,52588E-05
2	4	7,62939E-06
4	3	9,53674E-07
8	2	1,90735E-06
16	2	5,96046E-08
32	1	7,62939E-06
64	1	9,53674E-07
128	1	1,19209E-07
256	1	1,49012E-08
512	1	1,86265E-09
1024	1	2,32831E-10
2048	1	2,91038E-11

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Table 2

Results of comparison of floating point and fractional up_s-values computed

s	m	maximum absolute deviation
1	8	2,22044E-16
2	4	1,11022E-16
4	3	5,55111E-17
8	2	2,77555E-17
16	2	1,38777E-17
32	1	1,22311E-25
64	1	2,01948E-26
128	1	1,43211E-27
256	1	1,56672E-28
512	1	1,11022E-28
1024	1	1,08421E-19
2048	1	5,42101E-20

Finally, in Table 3, results of verification are given. Approach, which is introduced in the previous subsection, is applied. It can be concluded that maximum deviation is less than $2,22044\times10^{-15}$.

Table 3

Results of verification of up_s-values computed

s	m	type of values	maximum absolute deviation
1	8	floating point	2,22044E-16
		fractional	0
2	4	floating point	1,11022E-16
		fractional	0
4	3	floating point	5,55111E-17
		fractional	0
8	2	floating point	2,77555E-17
		fractional	0
16	2	floating point	1,38777E-17
		fractional	0
32	1	floating point	1,22311E-25
		fractional	0
64	1	floating point	2,01948E-26
		fractional	0
128	1	floating point	1,43211E-27
		fractional	0
256	1	floating point	1,56672E-28
		fractional	0
512	1	floating point	1,11022E-28
		fractional	0
1024	1	floating point	1,08421E-19
		fractional	0
2048	1	floating point	5,42101E-20
		fractional	0

5. Discussion of the results

Analyzing the results of numerical experiments, we see that fractional values of $up_s(x_{s,2m,j})$ have been computed precisely, and the floating point ones have been obtained with errors, which do not exceed 10^{-15}. Moreover, maximum absolute deviation between the corresponding values is not greater than 10^{-15}. Hence, the difference is insignificant.

Nevertheless, fractional values of up_s-function should be used especially in those cases, when accumulated errors significantly impair efficiency of the algorithms applied. At the same time, storing of fractional values requires a lot more memory than floating point ones.

Furthermore, computation of $up_s\left(x_{s,2m,j}\right)$ can be accelerated by applying of (18). Indeed, it follows from this formula that

$$up_s\left(x_{s,2m,(2s)^{2m}-j}\right)=\frac{1}{s}-up_s\left(x_{s,2m,j}\right)$$

for any $j = 0, 1, \ldots, s(2s)^{2m-1}-1$.

Also, we get

$$up_s\left(x_{s,2m,s(2s)^{2m-1}}\right)=\frac{1}{2s}.$$

Hence, values of the function $up_s(x)$ on the grid $X_{s,2m}$ can be found two times faster. But (18) cannot be used to verify computed values in this case.

For this purpose, other properties of up_s-functions can be applied. For instance, it was shown in [3, 12] that for any $s = 1, 2, \ldots$ and each $n = 0, 1, 2, \ldots$ there exists coefficients $\left\{c_{s,n,k}\right\}$, such that

$$\sum_{k=-\infty}^{\infty} c_{s,n,k}\cdot up_s\left(x-\frac{k}{(2s)^n}\right)\equiv x^n.$$

In other words, spaces of linear combinations of up_s-functions' shifts contain algebraic polynomials. This feature provides development of numerous different verification rules.

Their choice allows increasing trustworthiness of verification. Nevertheless, their implementation is more complicated.

Finally, it is obvious that some steps of the proposed dynamic algorithm can be easily parallelized [17, 18], which provides faster computing of $up_s(x)$.

Conclusions

In this paper, we have developed an algorithm for calculation of values of atomic functions $up_s(x)$ on dense grids. The algorithm proposed is based on dynamic programming principles that provide low complexity computing.

Moreover, values of up_s-functions can be obtained precisely, which allow reducing accumulated errors that occur, especially, when processing big data. Also, verification procedure, which has linear complexity, has been proposed.

It is expected that the results of the current research will improve existing data processing technologies based on atomic functions $up_s(x)$, especially discrete atomic compression of digital images, and accelerate development of the new ones for real time applications [21, 22], cloud based [23] and mobile [24] computing.

Future research and development steps can be dedicated to computation of parameters of discrete atomic transform, which is a core of the algorithm DAC, as well as construction of verification rules.

References (GOST 7.1:2006)

1. Рвачов, В. Л. Про одну фінітну функцію [Текст] / В. Л. Рвачов, В. О. Рвачов // Доповіді АН УРСР, Сер. А. – 1971. – № 8. – С. 705-707.

2. Rvachev, V. A. On approximation by means of the function up(x) [Text] / V. A. Rvachev // Sov. Math. Dokl. – 1977. –No. 18. – P. 340-342.

3. Рвачёв, В. Л. Неклассические методы теории приближений в краевых задачах [Текст] / В. Л. Рвачёв, В. А. Рвачёв. – К. : Наукова думка, 1979. – 196 с.

4. Rvachev, V. A. Compactly supported solutions of functional-differential equations and their applications [Text] / V. A. Rvachev // Russian Math. Surveys. – 1990. – Vol. 45, No. 1. – P. 87 – 120.

5. Gotovac, H. Adaptive Fup multi-resolution approach to flow and advective transport in highly heterogeneous porous media: methodology, accuracy and convergence [Text] / H. Gotovac, V. Cvetkovic, R. Andricevic // Adv. Water Resour. – 2009. – Vol. 32, No. 6. – P. 885-905.

6. Gotovac, H. Multi-resolution adaptive modeling of groundwater flow and transport problems [Text] / H. Gotovac, R. Andricevic, B. Gotovac // Adv. Water Resour. – 2007. – Vol. 30, No. 5. – P. 1105-1126.

7. Brysina, I. V. Discrete atomic compression of digital images [Text] / I. V. Brysina, V. O. Makarichev // Radioelectronic and Computer Systems. – 2018. – No. 4 (88). – P. 17-33. DOI: 10.32620/reks.2018.4.02.

8. Lukin, V. Discrete atomic compression of digital images: a way to reduce memory expenses [Text] / V. Lukin, I. Brysina, V. Makarichev // Integrated Computer Technologies in Mechanical Engineering. Advances in Intelligent Systems and Computing / V. Lukin, I. Brysina, V. Makarichev ; editors: M. Nechyporuk, V. Pavlikov, D. Kritskiy. – Springer, Cham, 2020. – Vol. 1113. – P. 492-502. DOI: 10.1007/978-3-030-37618-5_42.

9. Atomic wavelets in lossy and near-lossless image compression [Electronic resource] / V. Makarichev, V. Lukin, I. Brysina, B. Vozel, K. Chehdi // Proc. SPIE 11533, Image and Signal Processing for Remote Sensing XXVI. – 2020. – Vol. 11533. DOI: 10.1117/12.2573970.

10. Makarichev, V. Lossless Discrete Atomic Compression of Full Color Digital Images [Text] / V. Makarichev, V. Lukin, I. Brysina // Proc. 2021 IEEE 16th International Conference on the Experience of Designing and Application of CAD Systems (CADSM), Lviv, 22-26 February 2021. – 2021. – P. 43-46. DOI: 10.1109/CADSM52681.2021.9385239.

11. Рвачов, В. О. Деякі атомарні функції та їх застосування [Текст] / В. О. Рвачов, Г. О. Старець // Доповіді АН УРСР, Сер. А. – 1983. – № 11. – С. 22-24.

12. Makarichev, V. A. Approximation of periodic functions by $mup_s(x)$ [Text] / V. A. Makarichev // Math. Notes. – 2013. – Vol. 93, No. 6. – P. 858-880.

13. Старец, Г. А. Некоторые свойства функции $mup_n(x)$ [Текст] / Г. А. Старец // Математические методы анализа динамических систем. – 1983. – Вып. 7. – С. 15-17.

14. Старець, Г. О. Про моменти та значення деяких атомарних функцій [Текст] / Г. О. Старець, Л. І. Курпа // Системи озброєння і військова техніка. – 2010. – Вип. 3(23). – С. 162-163.

15. Старець, Г. О. Про атомарні функції $up_m(x)$ [Текст] / Г. О. Старець, І. І. Сидоренко // Системи управління, навігації та зв'язку. – 2008. – Вип. 2(6). – С. 175-177.

16. Introduction to Algorithms: Third Edition [Text] / T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein. – MIT Press, 2009. – 1292 p.

17. Pas, R. Using OpenMP – The Next Step: Affinity, Accelerators, Tasking, and SIMD [Text] / R. Pas, E. Stotzer, C. Terboven. – MIT Press, 2017. – 392 p.

18. Gropp, W. Using MPI: Portable Parallel Programming with the Message-Passing Interface [Text] / W. Gropp, E. Lusk, A. Skjellum. – MIT Press, 2014. – 336 p.

19. Kharchenko, V. Invariant-Based Safety Assessment of FPGA Projects: Conception and Technique [Electronic resource] / V. Kharchenko, O. Illiashenko, V. Sklyar // Computers. – 2021. – Vol. 10, No. 10. Article Id: 125. DOI: 10.3390/computers10100125.

20. Gregorics, T. A unified approach of program verification [Text] / T. Gregorics, Z. Borsi // Acta Univ. Sapientiae, Informatica. – 2017. – Vol. 9, No. 1. – P. 65-82. DOI: 10.1515/ausi-2017-0005.

21. Rao, M. V. G. Implementation of real time image processing system with FPGA and DSP [Text] / M. V. G. Rao, P. R. Kumar, A. M. Prasad // Proc. 2016 International Conference on Microelectronics, Computing and Communications (MicroCom), Durgapur, India, 23-25 January 2016. – 2016. – P. 1-4.

22. Li, C. Architecture-Aware Optimization Strategies in Real-time Image Processing [Text] / C. Li, S. Balla-Arabe, F. Yang. – Wiley-ISTE, 2017. – 177 p.

23. Pandey, N. K. A Review on Cloud based Image Processing Services [Text] / N. K. Pandey, M. Diwakar // Proc. 2020 7th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 12-14 March 2020. – 2020. – P. 108-112.

24. Thabet, R. Image processing on mobile devices: an overview [Text] / R. Thabet, R. Mahmoudi, M. H.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Bedoui // Proc. International Image Processing, Applications and Systems Conference, Sfax, Tunisia, 5-7 November 2014. – 2014. – P. 1-8.

References (BSI)

1. Rvachev, V. L., Rvachev, V. O. Pro odnu finitnu funkciyu [On a finite function]. *Proc. Ukr. SSR Acad. Sci, Ser. A.*, 1971, no. 8, pp. 705-707.

2. Rvachev, V. A. On approximation by means of the function up(x). *Sov. Math. Dokl*. 1977, no. 18, pp. 340–342.

3. Rvachev, V. L., Rvachev, V. A. *Neklassicheskie metody teorii priblizhenii v kraevykh zadachakh* [Non-classical methods of approximation theory in boundary value problems]. Kyiv, "Naukova dumka" Publ., 1979. 196 p.

4. Rvachev, V. A. Compactly supported solutions of functional-differential equations and their applications. *Russian Math. Surveys*, 1990, vol. 45, no. 1, pp. 87-120.

5. Gotovac, H., Cvetkovic, V., Andricevic, R. Adaptive Fup multi-resolution approach to flow and advective transport in highly heterogeneous porous media: methodology, accuracy and convergence. *Adv. Water Resour.*, 2009, vol. 32, no. 6, pp. 885-905.

6. Gotovac, H., Andricevic, R., Gotovac, B. Multi-resolution adaptive modeling of groundwater flow and transport problems. *Adv. Water Resour.*, 2007, vol. 30, vo. 5, pp. 1105-1126.

7. Brysina, I. V., Makarichev, V. A. Discrete atomic compression of digital images. *Radioelectronic and Computer Systems*, 2018, vol. 88, no. 4, pp. 17-33. DOI: 10.32620/reks.2018.4.02.

8. Lukin, V., Brysina, I., Makarichev, V. Discrete Atomic Compression of Digital Images: A Way to Reduce Memory Expenses. In: *Integrated Computer Technologies in Mechanical Engineering. Advances in Intelligent Systems and Computing*, Springer, Cham, 2020, vol. 1113, pp. 492-502. DOI: 10.1007/978-3-030-37618-5_42.

9. Makarichev, V., Lukin, V., Brysina, I., Vozel, B., Chehdi, K. Atomic wavelets in lossy and near-lossless image compression. *Proc. SPIE 11533, Image and Signal Processing for Remote Sensing XXVI*, 2020, vol. 11533. DOI: 10.1117/12.2573970.

10. Makarichev, V., Lukin, V., Brysina, I. Lossless Discrete Atomic Compression of Full Color Digital Images. *Proc. 2021 IEEE 16th International Conference on the Experience of Designing and Application of CAD Systems (CADSM)*, 2021, pp. 43-46. DOI: 10.1109/CADSM52681.2021.9385239.

11. Rvachev, V. O., Starets, G. O. Dejaki atomarni funkcii ta jih zastosuvannya [Some atomic functions and their applications]. *Proc. Ukr. SSR Acad. Sci, Ser. A.*, 1983, no. 11, pp. 22-24.

12. Makarichev, V. A. Approximation of periodic functions by $mup_s(x)$. *Math. Notes*, 2013, vol. 93, no. 6, pp. 858-880.

13. Starets, G. A. Nekotorye svoistva funkcii $mup_n(x)$ [Some properties of the function $mup_n(x)$]. *Mathematical methods of analysis of dynamic systems*, 1983, no. 7, pp. 15-17.

14. Starets, G. O., Kurpa, L. I. Pro momenty ta znachennia dejakyh atomarnyh funckij [About moments and values of some atomic functions]. *Systems of Arms and Military Equipment*, 2010, no. 3(23), pp. 162-163.

15. Starets, G. O., Sidorenko, I. I. Pro atomarni funkcii $up_m(x)$ [About atomic functions $up_m(x)$]. *Control, Navigation and Communication Systems*, 2008, no. 2(6), pp. 175-177.

16. Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C. *Introduction to Algorithms: Third Edition*, MIT Press, 2009, 1292 p.

17. Pas, R., Stotzer, E., Terboven, C. *Using OpenMP – The Next Step: Affinity, Accelerators, Tasking, and SIMD*, MIT Press, 2017, 392 p.

18. Gropp, W, Lusk, E., Skjellum, A. *Using MPI: Portable Parallel Programming with the Message-Passing Interface*, MIT Press, 2014, 336 p.

19. Kharchenko, V., Illiashenko, O, Sklyar, V. Invariant-Based Safety Assessment of FPGA Projects: Conception and Technique. *Computers*, 2021, vol. 10, no. 10, article id: 125. DOI: 10.3390/computers10100125.

20. Gregorics, T., Borsi, Z. A unified approach of program verification. *Acta Univ. Sapientiae, Informatica*, 2017, vol. 9, no. 1, pp. 65-82. DOI: 10.1515/ausi-2017-0005.

21. Rao, M. V. G., Kumar, P. R., Prasad A. M. Implementation of real time image processing system with FPGA and DSP. *Proc. 2016 International Conference on Microelectronics, Computing and Communications (MicroCom)*, 2016, pp. 1-4.

22. Li, C., Balla-Arabe, S., Yang, F. *Architecture-Aware Optimization Strategies in Real-time Image Processing.* Wiley-ISTE Publ., 2017. 177 p.

23. Pandey, N. K., Diwakar, M. A Review on Cloud based Image Processing Services. *Proc. 2020 7th International Conference on Computing for Sustainable Global Development (INDIACom)*, 2020, pp. 108-112.

24. Thabet, R., Mahmoudi, R., Bedoui, M. H. Image processing on mobile devices: an overview. *Proc. International Image Processing, Applications and Systems Conference*, Sfax, Tunisia, 5-7 November 2014, 2014, pp. 1-8.

Надійшла до редакції 28.08.2021, розглянута на редколегії 26.11.2021

ЗАСТОСУВАННЯ ДИНАМІЧНОГО ПРОГРАМУВАННЯ ПРИ ОБЧИСЛЕННІ ЗНАЧЕНЬ АТОМАРНИХ ФУНКЦІЙ

В. О. Макарічев, В. С. Харченко

Розглянуто спеціальний клас атомарних функцій (АФ), якими називають розв'язки з компактним носієм лінійних функціонально диференціальних рівнянь з постійними коефіцієнтами та лінійними перетвореннями аргументу. Ці функції використовуються у дискретному атомарному стисненні (ДАС) цифрових зо-

бражень. Алгоритм ДАС є алгоритмом стиснення з втратами якості та забезпечує кращі результати, ніж алгоритм JPEG, який де-факто є стандартом для стиснення цифрових фотографій. Використання значень АФ високої точності дозволить покращити цей алгоритм і забезпечити можливість розробки нових технологій аналізу та обробки даних. Важливим є розроблення алгоритму з низькою складністю для обчислення точних значень АФ. Їх точні значення у точках густих сіток є **предметом** дослідження. Безпосереднє використання зазначених формул призводить до повторного обчислення одних і тих самих виразів. Запропоновано підхід, що усуває цей недолік. **Метою** дослідження є розроблення алгоритму, основаного на формулах В. О. Рвачова та їх узагальненнях. Розв'язуються такі **завдання**: перетворити відповідні формули з метою зменшення кількості операцій та розробити процедуру верифікації значень АФ. Використовуються **методи** теорії АФ у поєднанні з принципами динамічного програмування. Запропоновано обчислювальну схему, а також динамічний алгоритм обчислення значень АФ у точках сіток з кроком, меншим будь-якого малого додатного наперед заданого числа. Запропоновано процедуру верифікації, що базується на властивостях АФ. Отримано такі **результати**: 1) розроблений алгоритм забезпечує більш швидке обчислення, ніж безпосереднє використання відповідних формул; 2) алгоритм надає можливість точного обчислення значень АФ; 3) розроблена процедура верифікації обчислених значень має лінійну за кількістю елементів складність. Розроблений алгоритм реалізовано мовою програмування Python. Також отримано набір таблиць значень АФ. **Висновки**: результати дослідження дозволять покращити алгоритми обробки даних з використанням АФ, зокрема, алгоритм ДАС, а також прискорять розробку нових технологій.

Ключові слова: атомарна функція; up-функція; динамічне програмування; верифікація; дискретне атомарне стиснення.

ПРИМЕНЕНИЕ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ ПРИ ВЫЧИСЛЕНИИ ЗНАЧЕНИЙ АТОМАРНЫХ ФУНКЦИЙ

В. А. Макаричев, В. С. Харченко

Рассмотрен специальный класс атомарных функций (АФ), которыми называют решения с компактным носителем линейных функционально дифференциальных уравнений с постоянными коэффициентами и линейными преобразованиями аргумента. АФ нашли свое применение в дискретном атомарном сжатии (ДАС) цифровых изображений. Алгоритм ДАС является алгоритмом сжатия с потерями качества и обеспечивает лучшие результаты, чем алгоритм JPEG, который де-факто является стандартом для сжатия цифровых фотографий. Использование значений АФ высокой точности позволит улучшить этот алгоритм, обеспечит возможность разработки новых технологий анализа и обработки данных. Цель исследований – разработка алгоритма низкой сложности для вычисления точных значений АФ. Точные значения АФ в точках густых сеток являются **предметом** исследования. Используются формулы В. А. Рвачева и их обобщения. Непосредственное применение этих формул приводит к повторному вычислению одних и тех же значений. Предложен подход, устраняющий этот недостаток. **Целью** является усовершенствование алгоритма, основанного на формулах В. А. Рвачева и их обобщениях. Решаются **задачи**: выполнить преобразования соответствующих выражений с целью уменьшения числа операций, а также разработать процедуру верификации значений АФ. Используются **методы** теории АФ в сочетании с принципами динамического программирования. Получена вычислительная схема, разработан динамический алгоритм вычисления значений АФ в точках сеток с шагом, который не превосходит любое наперед заданное положительное число. Предложена процедура верификации, основанная на свойствах АФ. Получены такие **результаты**: 1) предложенный алгоритм обеспечивает более высокую скорость вычисления; 2) этот алгоритм позволяет точно вычислять значения АФ; 3) разработанная процедура верификации найденных значений имеет линейную сложность. Алгоритм реализован на языке программирования Python. Получен набор таблиц значений АФ. **Выводы**: результаты данного исследования позволяют улучшить алгоритмы обработки данных, которые основаны на применении АФ, в частности, алгоритм ДАС, а также ускорят разработку новых технологий.

Ключевые слова: атомарная функция; up-функция; динамическое программирование; верификация; дискретное атомарное сжатие.

Макарічев Віктор Олександрович – канд. фіз.-мат. наук, докторант кафедри комп'ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.

Харченко Вячеслав Сергійович – д-р техн. наук, проф., зав. кафедри комп'ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.

Victor Makarichev – PhD in Physics and Mathematics, doctoral student, Department Computer Systems, Networks and Cybersecurity, National Aerospace University "Kharkiv Aviation Institute", Kharkiv, Ukraine, e-mail: victor.makarichev@gmail.com, ORCID: 0000-0003-1481-9132, Scopus Author ID: 41761910800.

Vyacheslav Kharchenko – Doctor of Technical Science, Professor, Head of Department Computer Systems, Networks and Cybersecurity, National Aerospace University "Kharkiv Aviation Institute", Kharkiv, Ukraine, e-mail: v.kharchenko@csn.khai.edu, ORCID: 0000-0001-5352-077X, Scopus Author ID: 22034616000.