UDC 004.383
DOI: 10.15587/2706-5448.2024.301309

Igor Kulyk,
Maryna Shevchenko,
Anatolii Melnyk,
Tetyana Protasova

DEVELOPMENT OF HIGH-SPEED ALGORITHM FOR BINOMIAL ARITHMETIC ADDITION

The object of research is the method and algorithm of arithmetic addition of binomial numbers generated by binary binomial counting systems. The lack of binomial arithmetic, in particular the operation of adding binary binomial numbers, in a certain way prevents their introduction into information systems and the construction of information and communication technologies based on them for combinatorial optimization, generation of combinatorial objects, data compression and encryption.

In the framework of the proposed approach, instead of operating with binomial coefficients, only operations with their upper and lower parameters are carried out. At the same time, the weighting coefficients of binary binomial numbers, which are added to each other, are represented in the form of two-component tuples. Taking this into account, this paper presents an algorithm for binomial arithmetic addition using dynamic arrays.

The main idea, which is included in the structure of the algorithm of binomial arithmetic addition based on dynamic arrays, is that the transition from a two-dimensional model of summation to a one-dimensional one is carried out. At the same time, only available, existing binomial coefficients are placed in the dynamic array. Accordingly, the search for binomial coefficients equal to or greater than the quantitative equivalent takes place in much smaller areas. In comparison with the algorithm based on matrix models, this quite significantly reduces the amount of time spent when performing the summation operation, and also reduces the requirements for the amount of memory required for placing two-component tuples of the assembly array.

In the course of the research, a several-fold decrease in the number of machine cycles required to search for the necessary elements in the dynamic array was practically confirmed. This leads to an increase in the performance of the presented algorithm of binomial arithmetic addition based on dynamic arrays. In turn, this leads to the acceleration of solving information tasks of combinatorial optimization, generation of combinatorial objects, data compression and encryption, for the solution of which the operation of adding binary binomial numbers is used.

Keywords: *binary binomial numbers, arithmetic addition, binomial arithmetic addition algorithms, dynamic array.*

Received date: 23.02.2024
Accepted date: 04.04.2024
Published date: 09.04.2024

How to cite
Kulyk, I., Shevchenko, M., Melnyk, A., Protasova, T. (2024). Development of high-speed algorithm for binomial arithmetic addition. Technology Audit and Production Reserves, 2 (2 (76)), 25–31. doi: https://doi.org/10.15587/2706-5448.2024.301309

1. Introduction

At the current stage of development of digital computing systems, positional heterogeneous computing systems are becoming increasingly widespread for solving informational tasks [1–3]. Factorial, Fibonacci and binomial number systems occupy a special place among them. In comparison with traditional degree systems of counting, such as binary, they have a more complex structure and such positive qualities as redundancy, built-in means of detecting errors, and the ability to generate combinatorial objects. This allows heterogeneous computing systems to successfully solve specialized tasks of reliable message transmission, end-to-end control of processed data, information compression, combinatorial optimization, implementation of interference-resistant machine arithmetic [4–6]. At the same time, when solving many specialized problems, unique characteristics can be achieved in terms of speed or fault tolerance of the equipment, which cannot be obtained on the basis of the binary counting system, unless additional consumable hardware or software is used.

A special place among non-homogeneous ones is occupied by structural binomial counting systems with a binary alphabet [4, 7]. This is due to the fact that they have greater redundancy, are able to generate combinatorial objects, the structure of which is related to combination numbers, and also simply form binary binomial numbers. The prospect of a wider application of binary binomial counting systems for the creation of new information technologies for data processing is determined by the newly obtained matrix models of binomial numbers and the matrix binomial arithmetic addition algorithm [8, 9]. The algorithm for arithmetic addition of binary binomial numbers based on matrix models presented in [10] is characterized by efficiency and ease of practical implementation, as it consists of fairly simple operations. But its disadvantage is low speed, which is explained by the need to perform cell search operations in very bulky areas of the matrix of binomial arithmetic addition.

Therefore, reducing the time for obtaining the result of the arithmetic addition of binary binomial numbers is an urgent task, as it will contribute to the acceleration of the solution of many information problems, for example, processing of binomial numerical information, combinatorial optimization, generation of common combination codes, etc.

The theoretical aim of this research paper is to minimize the time spent on arithmetic addition of binary binomial numbers. *In practical terms*, this means speeding up the solution of specialized data processing tasks through the creation of new information and communication technologies based on binary binomial counting systems.

The task of this scientific work is to build a faster algorithm for the arithmetic addition of binary binomial numbers, which will eventually make it possible to increase the performance of specialized devices or software tools that function on the basis of binary binomial numbers.

The object of research is the method and algorithm of arithmetic addition of binomial numbers generated by binary binomial counting systems. *The subject of research* is the process of transformations of weight coefficients of binomial numbers and the formation of transfer units from one binomial digit to another, which occur during the arithmetic addition of binary binomial numbers.

2. Materials and Methods

2.1. Method of arithmetic addition for binary binomial numbers. Binary binomial number systems belong to the class of heterogeneous structural systems [4, 7]. The numerical function of the binary (n,k)-binomial counting system looks as follows [8, 10]:

$$F_j = \operatorname{dec} X_j = \sum_{i=1}^{r} x_i C_{n-i}^{k-q_i}, \tag{1}$$

where n and k – parameters of the binary binomial number system; x_i – the binary binomial number $x_i \in \{0,1\}$; q_i – the sum of unit values x_i from the first digit to the $(i-1)$-th inclusively, $0 \le q_i \le k-1$.

Binary (n,k)-binomial numbers $X_j = x_1x_2...x_i...x_r$ must satisfy systems of code-forming constraints of the form [8, 10]:

$$((l=n-k)\wedge(x_r=0))\vee((q=k)\wedge(x_r=1)), \tag{2}$$

where l and q – the number of zeros and ones, respectively, in a binary (n,k)-binomial number. Binary (n,k)-binomial numbers X_j generated using constraint systems (2) have an uneven length $min(k, n-k) \le r \le n-1$. To carry out arithmetic addition, the binomial numbers $X_j = x_1x_2...x_i...x_r$ are brought to even numbers by adding binary zeros to the last r-th digit, until the total number of digits is $n-1$. Some non-uniform and their corresponding uniform (8,4)-binomial numbers, as well as their quantitative equivalents F_j are listed in the Table 1 (added zero bits are marked in gray).

Numerical function (1) demonstrates the presence of a complex functional relationship between the values of the weighting coefficients of the binomial number X_j, which makes it much more difficult to perform arithmetic operations on X_j. To take into account this connection, it is proposed to represent the binomial coefficients from the numerical function (1) through two-element tuples (α_i, Δ_i) [8, 10]:

$$C_{n-i}^{k-q_i} = C_{\beta_i}^{\alpha_i} = C_{\alpha_i+\Delta_i}^{\alpha_i},$$

where $\Delta_i = \beta_i - \alpha_i$ is called a combination of parameters. The ordered q-sample $S_{Xj} = ((\alpha_i, \Delta_i)^{(1)}, (\alpha_i, \Delta_i)^{(2)}, ..., (\alpha_i, \Delta_i)^{(q)})$ consists of tuples (α_i, Δ_i), which uniquely displays the binary binomial number X_j. On the basis of the obtained q-sample S_{Xj}, a (0,1)-matrix of a binary binomial number is constructed, which has the dimension $k \times (n-k)$. According to the coordinates $1 \le \alpha_I \le k$ in the columns and $0 \le \Delta_i \le n-k-1$ in the rows, a unit is entered in the corresponding cell, and zeros are written in the remaining cells (the far right column $\alpha = k$, the highest row $\Delta = n-k-1$). The basis for the (0,1)-matrix is the matrix of possible weighting coefficients, so the presence of a unit cell means the presence of a corresponding binomial coefficient. On the basis of (0,1)-matrices of binary binomial numbers being added, a matrix of binomial arithmetic addition, the so-called (0,11...1)-matrix, is constructed, with the addition of a zero column $\alpha = 0$ on the right. The zero column is intended to form a unit, which completes the creation of a unit row or sub-row. Obtaining such a final unit precedes the operation of transfer to the highest binomial digit [8, 10].

Table 1
Correspondence between some irregular, regular (8,4)-binomial numbers and their quantitative equivalents F_j

F_j	Irregular (8,4)-binomial numbers							Regular (8,4)-binomial numbers						
1	0	0	0	1	0			0	0	0	1	0	0	0
9	0	0	1	1	0	0		0	0	1	1	0	0	0
24	0	1	0	1	1	1		0	1	0	1	1	1	0
37	1	0	0	0	1	1	0	1	0	0	0	1	1	0
69	1	1	1	1				1	1	1	1	0	0	0

As an example, Fig. 1 shows the (0,1)-matrices of binary (8,4)-binomial numbers $X'_j = 0011000$ and $X''_j = 0101110$, as well as the appearance of the original (0,11...1)-matrix of binomial arithmetic addition.

$$\left\|X'_j\right\| = \begin{array}{|c|c|c|c|}\hline 0&0&0&0\\\hline 0&0&0&0\\\hline \mathbf{1}&\mathbf{1}&0&0\\\hline 0&0&0&0\\\hline\end{array} \qquad \left\|X''_j\right\| = \begin{array}{|c|c|c|c|}\hline 0&0&0&0\\\hline \mathbf{1}&0&0&0\\\hline 0&\mathbf{1}&\mathbf{1}&\mathbf{1}\\\hline 0&0&0&0\\\hline\end{array}$$

$$\left\|X'_j + X''_j\right\| = \begin{array}{|c|c|c|c|c|}\hline 0&0&0&0&0\\\hline \mathbf{1}&0&0&0&0\\\hline \mathbf{1}&\mathbf{11}&\mathbf{1}&\mathbf{1}&0\\\hline 0&0&0&0&0\\\hline\end{array}$$

Fig. 1. (0,1)-matrices of binary binomial numbers $X'_j = 0011000$, $X''_j = 0101110$ and the original (0,11...1)-matrix of arithmetic addition

In the (0,11...1)-matrix, a cell with several units is allowed, that is, the existence of several identical weighting coefficients is displayed.

To transform the cells of the assembly matrix, namely the coordinates α and Δ, in order to perform $X'_j + X''_j$, known combinatorial relations [9] for the numbers of combinations are used. The transformation of the binomial coefficients, expressed through the upper α and lower Δ parameters, looks as follows:

1) transformation V transfer:

$$(\alpha,\ \Delta)=(\alpha,\ \Delta-1)+(\alpha-1,\ \Delta-1)+\ \dots\ +$$
$$+(1,\ \Delta-1)+(0,\ \Delta-1); \qquad (3)$$

2) W shift transformation:

$$(\alpha,\ 0)=(\chi,\ 0)=(0,\ \Delta); \qquad (4)$$

3) B symmetry transformation:

$$(\alpha,\ \Delta)=(\Delta,\ \alpha); \qquad (5)$$

4) transformation D decomposition:

$$(\alpha,\ \Delta)=(\alpha,\ \Delta-1)+(\alpha-1,\ \Delta). \qquad (6)$$

Considering transformations (3)–(6), the main advantage of the binomial addition method based on matrix models is that operations are performed not with binomial coefficients, but with their parameters α and Δ. At the same time, these actions are limited only to comparison operations, changing parameter cities, and subtracting a unit.

In the algorithm presented in [10], the most time-consuming operation is the search for transformed cells $(\alpha_s,\ \Delta_s)$ to obtain the current $(\alpha_t,\ \Delta_t)$ in order to form from a series of units in (0,11...1)-matrices. This search takes place according to systems of equalities:

$$((0\le\alpha_s\le\alpha_t-1)\wedge(0\le\Delta_s\le\Delta_t+1))\vee((\alpha_t\le\alpha_s\le k)\wedge$$
$$\wedge(0\le\Delta_s\le\Delta_t-1)), \qquad (7)$$

when searching for equal $(\alpha_s,\ \Delta_s)=(\alpha_t,\ \Delta_t)$, or:

$$((0\le\alpha_s\le\alpha_t-1)\wedge(0\le\Delta_s\le\Delta_t+1))\vee((\alpha_t\le\alpha_s\le k)\wedge$$
$$\wedge(0\le\Delta_s\le\Delta_t)), \qquad (8)$$

when searching for a larger $(\alpha_s,\ \Delta_s)>(\alpha_t,\ \Delta_t)$.

Fig. 2 shows as an example in gray the search area (α_s, Δ_s) for finding an equal cell for the current (2, 3) in the process of adding binary binomial numbers 01101100 and 00101011.

0	0	0	0	0
1	1	[0]	0	0
1	0	1	1	0
0	1	0	0	0
0	0	1	1	0

Fig. 2. Demonstration of the search area when composing binary binomial numbers

Considering Fig. 2, it is possible to see that the search area $(\alpha_s,\ \Delta_s)$ contains 19 cells that need to be viewed. Accordingly, it is necessary to spend 19 machine cycles instead of processing only the available unit cells of the matrix of arithmetic addition. Therefore, it is proposed to minimize the time of the operation in such a way as to operate and search for the reqired $(\alpha_t,\ \Delta_t)$ only among single elements that reflect the available weighting factors. In this case, use a dynamic array of binomial numbers as a matrix of binomial addition. As a result, an algorithm for the arithmetic addition of binary binomial numbers is synthesized based on a dynamic array of two-element tuples (α, Δ) that reflect the parameters of binomial (weighting) coefficients.

2.2. Algorithm for binomial arithmetic addition based on dynamic arrays. The algorithm proposed below uses a dynamic array $S[(\alpha,\ \Delta),\ \gamma]$ of arithmetic composition, where γ – an index variable of the array. The arrangement of the elements (α, Δ) in the array is ordered as follows: with a fixed $0\le\Delta\le n-k-1$, starting with $\Delta=n-k-1$, all tuples $(k\le\alpha\le n,\ n-k-1)$, starting with $\alpha=k$. Adding elements to the array $S[(\alpha,\ \Delta),\ \gamma]$ according to the algorithm takes place in the specified order of placement. Let's also agree that the symbol «/» indicates the operation of removing elements from the array, the symbol «++» – the operation of adding elements to the array.

Step 1. Filling the dynamic arrays $X'[\mathrm{m}]$ and $X''[\mathrm{l}]$ with tuples $(\alpha'_\mu,\ \Delta'_\mu)$ and $(\alpha''_\lambda,\ \Delta''_\lambda)$ corresponding to binary (n,k)-binomial numbers X_j' and X_j'':

$$X'[\mu]=((k,\ \Delta'_1),\ (k-1,\ \Delta'_2),\ \dots,\ (\alpha'_\mu,\ \Delta'_\mu),$$
$$\dots,\ (k-q'+1,\ \Delta'_{q'})),\ \mu=1,\ 2,\ \dots,\ q';$$

$$X''[\lambda]=((k,\ \Delta''_1),\ (k-1,\ \Delta''_2),\ \dots,\ (\alpha''_\lambda,\ \Delta''_\lambda),$$
$$\dots,\ (k-q''+1,\ \Delta''_{q''})),\ \lambda=1,\ 2,\ \dots,q''.$$

Step 2. The assignment operation is performed:

$$S[(\alpha,\ \Delta),\ \gamma]=X'[\mu],\ S[(\alpha,\ \Delta),\ \gamma]=X''[\lambda],$$
$$\gamma=1,\ 2,\dots,\ \gamma_{\max},\ \gamma_{\max}=q'+q''.$$

There is the initial dynamic array of arithmetic addition to implement transformations V, W, B and D (3)–(6):

$$S[(\alpha,\ \Delta),\ \gamma]=((k,\ \Delta'_1),\ (k-1,\ \Delta'_2),$$
$$\dots,\ (\alpha_\gamma,\ \Delta_\gamma),\ \dots,\ (k-q''+1,\ \Delta''_{q''}))=$$
$$=((k,\ \Delta_1),\ (\alpha_2,\ \Delta_2),\ \dots,\ (\alpha_\gamma,\ \Delta_\gamma),\ \dots,\ (\alpha_{q'+q''},\ \Delta_{q'+q''})),$$

where the array can contain two or more identical tuples (α, Δ), at the beginning of the algorithm $\max\gamma\le 2k$.

Step 3. The initial element of the array $S[(\alpha,\ \Delta),\ \gamma]$ has the form:

$$(\alpha_t,\ \Delta_t)=S[(\alpha,\ \Delta),\ \gamma=1]=(k,\ \Delta_1),$$

where α_t and Δ_t – the current values of the parameters of the array element being processed.

Step 4. If for the elements (α, Δ) of the array $S[(\alpha, \Delta), \gamma]$ with $\gamma=1, 2, \dots, \gamma_{\max}$ and each fixed $\alpha=k,\ k-1,\ \dots,\ k-\gamma_{\max}+1$ the condition is fulfilled:

$$\exists!(\alpha,\ \Delta)\ \text{at}\ \Delta=n-k-1,\ n-k-2,\ \dots,\ 0,$$

then the transition to the next step is carried out (the first condition for the completion of binomial addition). Otherwise, proceed to Step 6.

Step 5. If for each pair of elements $(\alpha'+1,\ \Delta'')$ and $(\alpha',\ \Delta')$ of the array $S[(\alpha,\ \Delta),\ \gamma]$ of binomial composition with $\gamma=1,\ 2,\ \dots,\ \gamma_{\max}$ the condition is fulfilled:

$$\Delta''\ge\Delta'\ \text{at}\ 0\le\Delta',\ \Delta''\le n-k-1,$$

then the transition to Step 15 is carried out (the second condition for completion of binomial addition). Otherwise, proceed to Step 6.

Step 6. If for a given $0 \le \Delta \le n-k-1$ the condition is fulfilled:

$$\alpha_t = 0,$$

then the transition to the next step is performed (the fulfillment of this condition means the need to carry out the transfer to the higher binomial digit). Otherwise, proceed to Step 8.

Step 7. For a given $0 \le \Delta \le n-k-1$, elements are removed and added:

$$S[(\alpha, \Delta), \gamma] =$$
$$= S[(\alpha, \Delta), \gamma]/((\alpha_t, \Delta_t), (\alpha_t-1, \Delta_t), \ldots, (0, \Delta_t)),$$

$$S[(\alpha, \Delta), \gamma] = S[(\alpha, \Delta), \gamma] + +(\alpha_t, \Delta_t+1),$$

execution of the transformation V transfer (3). In this case, $\gamma_{\max} = \gamma_{\max} - \alpha_t$ and the initial element becomes: $(\alpha_t, \Delta_t) \leftarrow \leftarrow (\alpha_t-1, \Delta_t+1)$ (the initial element (α_t, Δ_t) can be part of the array binomial addition, and be absent).

Step 8. If the array $S[(\alpha, \Delta), \gamma]$ satisfies the condition:

$$(\exists(\alpha_t, \Delta_t)) \wedge (\exists(\alpha_t-1, \Delta_t+1)),$$

then operations of removing and inserting elements are carried out:

$$S[(\alpha, \Delta), \gamma] = S[(\alpha, \Delta), \gamma]/(\alpha_t, \Delta_t),$$

$$S[(\alpha, \Delta), \gamma] = S[(\alpha, \Delta), \gamma]/(\alpha_t-1, \Delta_t+1),$$

$$S[(\alpha, \Delta), \gamma] = S[(\alpha, \Delta), \gamma] + +(\alpha_t, \Delta_t+1),$$

inverse transformation D^{-1} of expansion (6). At the same time, $\gamma_{\max} = \gamma_{\max} - 1$ and the initial element becomes: $(\alpha_t, \Delta_t) \leftarrow \leftarrow (\alpha_t, \Delta_t+1)$. Then proceed to Step 9. Otherwise, proceed to the next step:

Step 9. If the condition is satisfied for the array $S[(\alpha, \Delta), \gamma]$:

$$\exists(\alpha_t, \Delta_t),$$

then the initial element becomes: $(\alpha_t, \Delta_t) \leftarrow (\alpha_t-1, \Delta_t)$ and the transition to Step 9 is performed. Otherwise, the transition to the next step is performed (formation of a series of sequentially placed elements (α, Δ) in the array at a fixed $0 \le \Delta_t \le n-k-1$, starting from the value $1 \le \alpha_t \le k$).

Step 10. If in the component value change area:

$$((0 \le \alpha_s \le \alpha_t-1) \wedge (0 \le \Delta_s \le \Delta_t+1)) \vee ((\alpha_t \le \alpha_s \le k) \wedge$$
$$\wedge (0 \le \Delta_s \le \Delta_t-1)),$$

of array elements $S[(\alpha, \Delta), \gamma]$ there is such an element, which component values satisfy the following condition:

$$((\Delta_s = 0) \wedge ((\Delta_t = 0)) \vee (\alpha_t = 0)) \vee ((\alpha_t = \Delta_s) \wedge (\Delta_t = \alpha_s)),$$

then operations of removing and inserting elements are carried out:

$$S[(\alpha, \Delta), \gamma] = S[(\alpha, \Delta), \gamma]/(\alpha_s, \Delta_s),$$

$$S[(\alpha, \Delta), \gamma] = S[(\alpha, \Delta), \gamma] + +(\alpha_t, \Delta_t),$$

where the search for the transformed element: the equality of the transformed and searched elements, only transformations of W shift (4) or B symmetry (5) are possible. At the same time, the maximum value $\gamma_{\max}$ of the array remains unchanged and the transition to Step 4 is performed. Otherwise, the transition to the next step is performed.

Step 11. If the elements of the array $S[(\alpha, \Delta), \gamma]$ fulfill the condition:

$$\exists(\alpha_t+1, \Delta_g), \text{ where } \Delta_g = \Delta_t+1, \Delta_t+2, \ldots, n-k-1,$$

then the transition is made to Step 13. Otherwise, the transition to the next step is made (search for the transformed element: the weight of the transformed element is greater than the weight of the sought one, the elements that already participate in the formation of series of sequentially located elements are taken into account at the values $\alpha = \alpha_t+1$ and $\Delta \ge \Delta_t+1$).

Step 12. If in the area of changing the values of the component:

$$((0 \le \alpha_s \le \alpha_t-1) \wedge (0 \le \Delta_s \le \Delta_t+1)) \vee ((\alpha_t \le \alpha_s \le k) \wedge$$
$$\wedge (0 \le \Delta_s \le \Delta_t-1)),$$

of the array elements $S[(\alpha, \Delta), \gamma]$ there is such an element for the value of which the condition is fulfilled:

$$(\alpha_s, \Delta_s) > (\alpha_t, \Delta_t),$$

then the transition to Step 14 is carried out (the weight of the detected element is greater than the weight of the sought one in the absence of elements having $\alpha = \alpha_t+1$ and $\Delta \ge \Delta_t+1$, the search for elements larger in terms of their quantitative equivalent in the above area means the presence of already formed $(k, \Delta_t), (k-1, \Delta_t), \ldots, (\alpha_t+1, \Delta_t)$). Otherwise, the initial element becomes $(\alpha_t, \Delta_t) \leftarrow (\alpha_t, \Delta_t-1)$ and the transition to Step 8 is made.

Step 13. If in the value change area of the component:

$$((0 \le \alpha_s \le \alpha_t-1) \wedge (0 \le \Delta_s \le \Delta_t+1)) \vee ((\alpha_t \le \alpha_s \le k) \wedge$$
$$\wedge (0 \le \Delta_s \le \Delta_t)),$$

of the array elements $S[(\alpha, \Delta), \gamma]$ there is such an element for the value of which the condition is fulfilled:

$$(\alpha_s, \Delta_s) > (\alpha_t, \Delta_t),$$

then the transition to the next step is carried out (the weight of the detected element is greater than the weight of the searched one in the presence of elements having $\alpha = \alpha_t+1$ and $\Delta \ge \Delta_t+1$, the search for elements larger in terms of their quantitative equivalent in the above area means the presence of forbidden $(k, \Delta_t), (k-1, \Delta_t), \ldots, (\alpha_t+1, \Delta_t)$). Otherwise, the initial element becomes $(\alpha_t, \Delta_t) \leftarrow (\alpha_t, \Delta_t-1)$ and the transition to Step 8 is made.

Step 14. Removal and insertion of elements are performed:

$$S[(\alpha, \Delta), \gamma] = S[(\alpha, \Delta), \gamma]/(\alpha_s, \Delta_s),$$

$$S[(\alpha, \Delta), \gamma] = S[(\alpha, \Delta), \gamma] + +((\alpha_s, \Delta_s-1), (\alpha_s-1, \Delta_s)).$$

At the same time, $\gamma_{\max} = \gamma_{\max}+1$ and the transition to Step 8 is carried out (transformation Δ of expansion (6) for the transformed element of the array).

Step 15. As a result, the array $S[(\alpha, \Delta), \gamma]$ takes the form of a dynamic array $Z[\varphi]$ of the result of the sum Z_j:

$$S[(\alpha, \Delta), \gamma]=((k, \Delta_1), (k-1, \Delta_2), ..., (\alpha_\varphi, \Delta_\varphi), ..., (k-q+1, \Delta_q))=Z[\varphi], \varphi=1, 2, ..., q; \gamma_{max}=q,$$

completion of binomial arithmetic addition $Z_j=X_j'+X_j''$ using a dynamic array of parameters of weighting coefficients.

Step 16. According to each $(\alpha_\varphi, \Delta_\varphi)$ of the obtained array $Z[\varphi]$, the unit digit number of the $(n-1)$-digit (n,k)-binomial number Z_j is calculated:

$$i=n-\alpha_\varphi-\Delta_\varphi.$$

The remaining digits are filled with zeros and the algorithm is completed (transition from the result array to the binary even (n,k)-binomial number Z_j).

3. Results and Discussions

Let's demonstrate the operation of the presented algorithm using the example of adding two (7,3)-binomial even numbers $X_j'=010010$ and $X_j''=100010$. At the same time, let's focus on the results of performing the main steps that lead to a change in the composition of the array $S[(\alpha, \Delta), \gamma]$ of binomial arithmetic addition.

At the beginning of the algorithm, that is, after Steps 1 and 2, there are the arrays corresponding to binomial numbers $X'[\mu]=((3, 2), (2, 0))$, $\mu=1, 2$ and $X''[\lambda]=((3, 3), (2, 0))$, $\lambda=1, 2$, as well as the array itself of the form $S[(\alpha, \Delta), \gamma]=((3, 3), (3, 2), (2, 0), (2, 0))$.

After Steps 3, 4, 6, 8, and 9, the element (2, 3) becomes the output. Then, in Step 10, let's find an element (3, 2) equal to it in quantitative value in the array $S[(\alpha, \Delta), \gamma]$. According to the transformation B of symmetry (5), the following operations are performed: $S[(\alpha, \Delta), \gamma]=S[(\alpha, \Delta), \gamma]/(3, 2)$ and $S[(\alpha, \Delta), \gamma]=S[(\alpha, \Delta), \gamma]++(2, 3)$, and the array itself looks as follows $S[(\alpha, \Delta), \gamma]=((3, 3), (2, 3), (2, 0), (1, 0))$. After Step 6 checks the formation of the series, Steps 8–13 are cyclically performed four times. The result of such cyclic execution is the assignment of the original current element (1, 0), searching and finding through Step 10 the corresponding quantitatively equivalent element (2, 0). Step 10 performs the operations $S[(\alpha, \Delta), \gamma]=S[(\alpha, \Delta), \gamma]/(2, 0)$ and $S[(\alpha, \Delta), \gamma]=S[(\alpha, \Delta), \gamma]++(1, 0)$, and the array itself appears as $S[(\alpha, \Delta), \gamma]=((3, 3), (2, 3), (2, 0), (1, 0))$. Next, the reference to Steps 4, 6 and 8, the conditions of which are not fulfilled, and the action of Step 9 assigns the original element (0, 0). The next Step 11 detects the corresponding quantitatively equivalent element (2, 0), which is contained in the array, and the operations $S[(\alpha, \Delta), \gamma]=S[(\alpha, \Delta), \gamma]/(2, 0)$ and $S[(\alpha, \Delta), \gamma]=S[(\alpha, \Delta), \gamma]++(0, 0)$, take place, and the array itself takes the form $S[(\alpha, \Delta), \gamma]=((3, 3), (2, 3), (1, 0), (0, 0))$. The algorithm termination condition in Step 4 is not satisfied because there is an element with $\alpha=0$. The next Step 6 detects a series of elements (1, 0), (0, 0) with a fixed $\Delta=0$, so Step 7 performs V transformation (3): $S[(\alpha, \Delta), \gamma]=S[(\alpha, \Delta), \gamma]/((1, 0), (0, 0))$ and $S[(\alpha, \Delta), \gamma]=S[(\alpha, \Delta), \gamma]++(1, 1)$. Then the addition array looks like $S[(\alpha, \Delta), \gamma]=((3, 3), (2, 3), (1, 1))$. The following Steps 4 and 5, checking the current form $S[(\alpha, \Delta), \gamma]$, determine the completion of binomial arithmetic addition. After Steps 15 and 16, there is the final resulting array $Z[\varphi]=((3, 3), (2, 3), (1, 1))$ and the corresponding binary (7,3)-binomial uniform number $Z_j=110010$.

Applying the numerical function (1) for the binary (7,3)-binomial number system, let's make sure that the obtained result is correct. Thus, the decimal equivalents of the binary binomial terms are *dec(010010)*=11 and *dec(100010)*=21, and the result of the binomial sum is *dec(110010)*=32. Hence, summing up the decimal values of the terms, there is the necessary result, which confirms the correctness of the binomial arithmetic addition algorithm.

A comparative analysis of the maximum values of the number of machine cycles spent on the search for array elements according to the presented algorithm or matrix cells according to the algorithm described in [10] is given in Table 2. For the initial array elements (matrix cells) of the form (1, 3), (1, 2) and (1, 1) in Table 2 shows the amount of clocks for searching both equal and greater than the quantitative equivalent of array elements (matrix cells). It should be noted that in Table 2 for the algorithm of binomial arithmetic addition based on matrix models, the maximum values of the number of cycles are indicated. The actual values of the numbers for it may be smaller, but with a high probability, the number of machine cycles required to find the corresponding elements in the case of applying an algorithm based on a dynamic array is still greater.

The ratio of the number of cycles used for the search depends on the technical features of the implementation of the algorithms. But it can be unequivocally noted that the algorithm of binomial arithmetic addition based on dynamic arrays has a potentially higher speed compared to the algorithm using matrix models.

Considering the systems of equalities (7) and (8), which indicate the search areas for cells (α_t, Δ_t) of the matrix of the addition of binary binomial numbers, it can be asserted that the maximum number of cycles spent for the operation is:

$$T'_{max}=\alpha_t\times(\Delta_t+2)+(k-\alpha_t+1)\times\Delta_t+ +\alpha_t\times(\Delta_t+2)+(k-\alpha_t+1)\times(\Delta_t+1). \quad (9)$$

Table 2

Analysis of the number of machine cycles when performing search operations

Maximum number of beats	When searching for an equal or greater quantitative equivalent of the current array element or matrix cell of the form					
	(2, 3)	(1, 3)	(1, 2)	(1, 1)	(1, 0)	(0, 0)
Algorithm based on the addition matrix [10]	14	26	23	15	5	5
Algorithm based on dynamic array	3	4	4	4	2	1
The ratio of the number of tacts to search, times	4.67	6.5	5.75	3.75	2.5	5

Opening the parentheses and summing the same terms in expression (9), the following simplified equality can be obtained for the $T'_{\max}$ value:

$$T'_{\max}=\alpha_t+(k+1)\times(2\Delta_t+1). \quad (10)$$

The maximum number of cycles $T''_{\max}$ for finding the element (α_t, Δ_t) of the dynamic array with a high probability will not exceed the double value of the maximum number of binary units $2k$ in the binomial number, that is, it is possible to assume:

$$T''_{\max}=2k. \quad (11)$$

This is the case when two binomial numbers are added, each of which contains $q=k$ units. Comparing expressions (10) and (11) for $T'_{\max}$ and $T''_{\max}$, respectively, it is possible to conclude that $T'_{\max}\geq T''_{\max}$ at $\Delta_t=1,2, ..., n-k-1$ and fixed $\alpha_t=0, 1, ..., k$. A more accurate assessment requires taking into account the technical properties of implementing binomial addition algorithms based on matrix models and dynamic arrays.

The advantages of the algorithm based on dynamic arrays presented in the work are the following:

1. Storage and operation only with the available elements of the dynamic array of binomial arithmetic addition leads to a significant increase in the speed of the summation algorithm of binary binomial numbers. In turn, this will have a positive effect on the acceleration of the solution of various information problems of combinatorial optimization, data compression and encryption, etc., in which binary binomial numbers are used.
2. The transition to the use of a dynamic array, in which only the available elements are placed during transformations of weighting coefficients, will allow to reduce the amount of RAM used to store the array of binomial arithmetic addition. In general, this leads to a decrease in the total volume of hardware and software costs and the cost of practical implementation of the arithmetic addition of binary binomial numbers.
3. On the basis of the presented method and algorithms for the arithmetical composition of binary binomial numbers, it becomes possible to create and implement new effective information and communication technologies for information compression, interference-resistant coding and encryption of data, and combinatorial optimization.

The disadvantages of the binomial arithmetic addition algorithm based on dynamic arrays considered in the work are the following:

1. Compared to the algorithm based on matrix models, the operations with the elements of the dynamic assembly array are more complex, in particular the operations of extracting, inserting and arranging elements.
2. In comparison with the algorithm based on matrix models, the procedures for current control of the correctness of intermediate transformations of weighted binomial coefficients are more complicated.
3. The transition from matrix placement of tuples (α, Δ) to one-dimensional in a dynamic array leads to a more complex organization of the control system by arithmetic addition of binary binomial numbers. As a result, there is a certain increase in hardware and software costs when implementing the control unit of the binomial arithmetic addition system.

The possibilities and prospects of further research on binomial arithmetic addition are as follows:

1. Development of rules and procedures for the arithmetic addition of binary (n,k)-binomial numbers in the case when there are transfers outside the matrix or array of binomial addition at given parameters n and k, i. e. outside the range of binary binomial numbers.
2. Development of new information and communication technologies using the method and algorithms of binomial arithmetic composition for solving problems of combinatorial optimization, data compression, interference-resistant coding and encryption of information.

The presented algorithm has a special prospect of use for solving specialized information tasks with limitations on time costs. For example, such specialized tasks include the generation of combination codes, arithmetic processing of signals represented by balanced codes, information compression using binomial numbers. Binary binomial numbers are the basis of the structure of many combinatorial sequences (equilibrium or quasi-equilibrium codes, codes with restrictions on the placement of ones or zeros). Having an additional effective mechanism in the form of a method and a fast-acting algorithm for the arithmetic addition of binomial numbers, it is possible to perform computational operations already on the specified combinatorial sequences. This significantly expands the functionality of information transformations of combinatorial codes. It should also be noted that the method and algorithms of binomial arithmetic addition did not exist until now.

A certain hindering factor regarding the implementation of the method and fast algorithm of binomial arithmetic addition is the complexity of binary binomial counting systems and the binomial numbers generated by them. Accordingly, this leads to the complication of the structure and practical implementation of binomial hardware and/or software systems, which limits their use for solving common information tasks. Binary binomial numbers show the greatest effectiveness in solving specialized problems of generating combinatorial objects built on the basis of combination codes, binomial compression and encryption of binary data.

The influence of martial law conditions. The experience of operating the electronic equipment of the military equipment of the armed forces of Ukraine demonstrates the extreme importance of protecting the built-in control and data transmission systems, in particular, unmanned aerial vehicles, from external influences. This ensures the necessary levels of digital equipment fault tolerance and data exchange immunity. Based on the immunity inherent in binary binomial numbers and their ability to quickly solve combinatorial problems of recognition and finding optimal solutions, the task of developing binomial arithmetic becomes especially urgent. Under the conditions of martial law in Ukraine, this will accelerate the construction of protected information and communication technologies on its basis for implementation in military information systems.

4. Conclusions

The developed algorithm of binomial arithmetic addition using dynamic arrays, in comparison with the algorithm based on matrix models, allows to reduce the time spent on the operation of adding numbers. In addition, during the practical implementation of the presented algorithm, the requirements for the required amount of RAM required for placing and saving the elements of the dynamic array are reduced. Saving time and hardware and software costs is due to the transition from a two-dimensional model of binomial arithmetic addition to a one-dimensional array, which is built only on the basis of available binomial coefficients.

The obtained high-speed algorithm of binomial arithmetic addition provides additional opportunities in the creation of new information and communication technologies for combinatorial optimization, generation of combinatorial objects, binomial compression and encryption of binary data. The effectiveness of solving the specified specialized tasks is due to the use of binary binomial counting systems and the binomial numbers generated by them.

Conflict of interest

The authors declare that they have no conflict of interest concerning this research, whether financial, personal, authorship or otherwise, that could affect the study and its results presented in this paper.

Financing

The study was performed without financial support.

Data availability

The paper has no associated data.

Use of artificial intelligence

The authors confirm that they did not use artificial intelligence technologies when creating this work.

References

1. Stakhov, A. P. (2014). A History, the Main Mathematical Results and Applications for the Mathematics of Harmony. *Applied Mathematics, 5 (3),* 363–386. doi: https://doi.org/10.4236/am.2014.53039
2. Borysenko, O. A. (2007). Chyslo i systemy chyslennia v elektronnykh tsyfrovykh systemakh. *Visnyk SumDU, 4,* 71–76.
3. Butler, T. J., Tsutomu, S. (1997). Redundant Multiple-Valued Number Systems: Report. Defense Technical Information center. *Naval Postgraduate School Monterey CA Dept of Electrical and Computer engineering,* 10. Available at: https://apps.dtic.mil/sti/pdfs/ADA599946.pdf Last accessed: 10.02.2024
4. Borisenko, A. A. (2004). *Binomialnyi schet. Teoriia i praktika.* Sumy: ITD «Universitetskaia kniga», 170.
5. Mezmaz, M., Leroy, R., Melab, N., Tuyttens, D. (2014). A Multicore Parallel Branch-and-Bound Algorithm Using Factorial Number System. *2014 IEEE 28th International Parallel and Distributed Processing Symposium.* Phoenix, 1203–1212. doi: https://doi.org/10.1109/ipdps.2014.124
6. Cui, X., Cui, X., Ni, Y., Miao, M., Yufeng, J. (2017). An Enhancement of Crosstalk Avoidance Code Based on Fibonacci Numeral System for Through Silicon Vias. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 25 (5),* 1601–1610. doi: https://doi.org/10.1109/tvlsi.2017.2651141
7. Borysenko, O., Matsenko, S., Bobrovs, V. (2021). Binomial Number System. *Applied Sciences, 11 (23),* 11110. doi: https://doi.org/10.3390/app112311110
8. Kulik, I. A., Shevchenko, M. S., Grinenko, V. V. (2022). Algoritm skladannia dviikovikh binomialnikh chisel. *Sistemi obrobki informatcii, 2 (169),* 49–57.
9. Kulyk, I. A., Shevchenko, M. S. (2021). Matrychna model skladannia dviikovykh binomialnykh chysel. *Systemy obrobky informatsii, 1 (164),* 45–54.
10. Anderson, Ja. A. (2001). *Discrete mathematics with combinatorics.* Prentice-Hall, Inc., 960.

✉***Igor Kulyk,*** *PhD, Associate Professor, Department of Electronics and Computer Technics, Sumy State University, Sumy, Ukraine, e-mail: i.kulyk@ekt.sumdu.edu.ua, ORCID: https://orcid.org/0000-0003-2403-8671*

Maryna Shevchenko, *PhD, Assistant, Department of Electronics and Computer Technics, Sumy State University, Sumy, Ukraine, ORCID: https://orcid.org/0000-0002-1434-5996*

Anatolii Melnyk, *Senior Research Fellow, Research Center of Missile Troops and Artillery, Sumy, Ukraine, ORCID: https://orcid.org/0000-0002-8466-9718*

Tetyana Protasova, *Senior Lecturer, Department of Electronics and Computer Technics, Sumy State University, Sumy, Ukraine, ORCID: https://orcid.org/0000-0002-8849-2665*

✉*Corresponding author*