Advances in Mathematics: Scientific Journal **9** (2020), no.12, 10419–10430
ISSN: 1857-8365 (printed); 1857-8438 (electronic)
https://doi.org/10.37418/amsj.9.12.31

ABOUT THE PROBLEM OF MINIMAL TESTS SEARCHING

ANVAR KABULOV[1], IBROKHIMALI NORMATOV, ERKIN URUNBAEV, AND AZIZ ASHUROV

ABSTRACT. The paper proposes algorithms for constructing minimal (dead-end) tests, testers, when solving problems of recognition and prediction. When obtaining algorithms, procedures were used to decode and search for the maximum upper zero of monotone Boolean functions; and the methods for solving problems using these procedures were given. To achieve the goals set in the paper, methods for solving the problems of decoding and searching for the maximum upper zero of mono-tone Boolean functions were studied, and an approximate parametric algorithm for solving these problems was constructed

1. INTRODUCTION

Solving the problems of recognition and prediction by test search methods, two types of problems are posed, basically: the synthesis of all or almost all dead-end tests and the construction of minimal tables.

V.E. Kuznetsov [1] and E.A. Dyukova in [3] described in sufficient detail the solutions to the problems of the first type. The problems of the second type, met when minimizing disjunctive normal forms (d.n.f.) of logical functions, searching for a minimum joint subsystem of systems of equations, etc., are the aims of this study. Numerical methods for the synthesis of minimal tests were also developed.

[1] *corresponding author*

2020 *Mathematics Subject Classification.* 94C11, 03E40, 90C90.

Key words and phrases. Algorithm, pattern recognition, features, minimal tests, dead-end tests, testers, predictions, monotone Boolean functions, maximum upper zero, set, approximate parametric algorithm.

2. Statement of the problem.

Let a table 1 of elements be given, consisting of m rows (objects) and n columns (features).

TABLE 1. Table of elements

	x_1	x_2	$\ldots$	x_n
S_1	a_{11}	a_{12}	$\cdots$	a_{1n}
S_2	a_{21}	a_{21}	$\cdots$	a_{2n}
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
S_m	a_{m1}	a_{m2}	$\cdots$	a_{mn}

Here $a_{ij} \in \{0, 1, \ldots, k-1\}$ $k \geq 2$, $i = \overline{1, m}$, $j = \overline{1, n}$. The set of columns $M = \{x_{i_1}, x_{i_2}, \ldots, x_{i_k}\}$ of Table 1 is called a test if for any pair of rows S_i and S_j exists such $x_i \in M$, that $a_{it} \neq a_{jt}$.

A test is called a dead-end one if, after deleting any column x_i from it, it ceases to be a test. A test is called minimal if it contains the least number of columns among all tests in this table. It is known that the following statements were proven in [2, 3].

Theorem 2.1. *If in table 1* $\frac{m(n)}{2^{n/2}} \to 0$ *as* $n \to \infty$ *with probability 1, then arbitrarily selected* $r_2 = 2\log_k m + \sqrt{\log_k m}$ *columns of table 1 form a test.*

Theorem 2.2. *For the number* r_1 *of columns in Table 1 that form a test, the following estimate of* $r_1 \geq]\log_k m[+ 1$ *is valid.*

Denote by Ω the class of all sub-sets of the set of columns $\{x_{i_1}, x_{i_2}, \ldots, x_{i_k}\}$. It is easy to see that $|\Omega| = 2^n$, where $|M|$ is the cardinality of the set. Each subset

$$\omega = \{x_{i_1}, x_{i_2}, \ldots, x_{i_k}\} \in \Omega$$

is mutually - unambiguously compared to the Boolean set $\overline{\alpha_\omega} = (\alpha_1, a_2, \ldots, \alpha_n)$ from E_n^2, the unit coordinates of which are $\alpha_{i_2}, \alpha_{i_2}, \ldots, \alpha_{i_k}$. Here E_n^2 is the class of binary sets of length n.

Introduce a Boolean function $g\,(y_1, y_2, \ldots, y_n)$.

$$g\left(\alpha_1,\alpha_2,\ldots,\alpha_n\right)=\begin{cases}1, \text{ if } \omega\in\Omega \text{ in the table} \\ \text{corresponds to the set} \\ \left(\alpha_1,\alpha_2,\ldots,\alpha_n\right) \text{ and forms} \\ \text{test;} \\ 0, \text{ if not.}\end{cases}$$

If a set of variables ω is a test, then any subset $\omega'\in\Omega$ is such that $\omega\subseteq\omega'$ is a test. That's why $\tilde{\alpha}_{\omega'}\geq\tilde{\alpha}_{\omega}$ and $g\left(\tilde{\alpha}_{\omega'}\right)\geq g\left(\alpha_{\omega}\right)$.

The converse is also true: if ω is a test, then all $\omega'\in\Omega$ are such that $\omega\subseteq\omega'$ is not a test. In this case, $\tilde{\alpha}_{\omega'}\in\alpha_{\omega}$ and $g\left(\tilde{\alpha}_{\omega'}\right)\leq g\left(\tilde{\alpha}_{\omega}\right)$. Consequently, the monotonicity of the function $g\left(\tilde{y}\right)$ is proven.

Assume that $g\left(\tilde{y}\right)$ corresponds to table 1.

Consider the class M_n of monotone functions of Boolean algebra in n variables. A set $\tilde{\alpha}\in E_n^2$ is called the lower unit of function $f\in M_n$ if $f\left(\tilde{\alpha}\right)=1$ and for any β it is such that $\tilde{\beta}\leq\alpha$, $f\left(\beta\right)=0$.

The number of units in the set $\tilde{\alpha}\in E_n^2$ is called the level and denoted by $|\tilde{\alpha}|$, and the class of all sets of the level i, $i=\overline{0,n}$, by U_i-th level in E_n^2.

The lower unit $\tilde{\alpha}$ of function f is called its minimal lower unit (m.l.u.), if for any lower unit $\tilde{\beta}$ the function is $f\ |\tilde{\alpha}|\leq\left|\tilde{\beta}\right|$.

Assume that sets $\tilde{\alpha}=\left(\alpha_1,\alpha_2,\ldots,\alpha_n\right)$ of levels U_k in E_n^2, $k=\overline{0,n}$ are in lexicographic order [4], if they are arranged in U_k in ascending order of values $\sum\limits_{i=1}^{n}\alpha_i 2^{n-i}$. Let us assume that the set $\tilde{\beta}=\left(\beta_1,\beta_2,\ldots,\beta_n\right)$ immediately follows $\tilde{\alpha}=\left(\alpha_1,\alpha_2,\ldots,\alpha_n\right)$ in U_k, $k=\overline{0,n}$ if $B=\sum\limits_{i=1}^{n}\beta_i 2^{n-i}>A=\sum\limits_{i=1}^{n}\alpha_i 2^{n-i}$, and there is no such γ in U_k, that $B=\sum\limits_{i=1}^{n}\gamma_i 2^{n-i}>A$.

Obviously m.l.u. of function $g\left(\tilde{y}\right)$ in one-to-one manner corresponds to the minimum tests of table 1. Therefore, it is easy to see that the problem of synthesizing the minimum test of table 1 is identical to the problem of search-ing m.l.u. of function $g\left(\tilde{y}\right)$. The function $g\left(\tilde{y}\right)$ is set by the opera-tor A_g, which calculates a value of $g\left(\tilde{y}\right)$ by any set $\tilde{\alpha}$ of length n, that is, for any sub-set $\omega\subset\Omega$ corresponding to the set $\tilde{\alpha}$, it is calculated whether ω forms a test or not.

N.N. Katerinochkina in [5–8] solved the problem of searching for the maximum upper zero (m.u.z.) $f\in M_n$ in Shannon's statement. It should be noted that it is a dual problem to the task of searching for m.l.u. of function f.

Considering the tests properties and based on the conclusions of theorems 2.1, 2.2, the search algorithm of m.l.u. of function $g(\tilde{y})$ can be written in stages as follows.

The first stage. Let us count the number $Q(i)$ of different pairs $(\alpha_{ij}, \alpha_{ik})$, $j \neq k$, $j,\ k = \overline{1,m}$ in the i-th column of table 1 in ascending order $Q(i)$, $i = \overline{1,n}$. Let $i_1, i_2, \ldots, i_n$ be an enumeration of columns in a given order, that is $Q(i_1) \leq Q(i_2) \leq \ldots \leq Q(i_n)$. It is clear that after the first stage in table 1 the columns were rearranged as

$$\left\{ \begin{array}{ccccc} 1 & 2 & 3 & \ldots & n \\ i_1 & i_2 & i_3 & \ldots & i_n \end{array} \right\}.$$

The second stage. In a set E_n^2, we divide the sets by levels $V_0, V_1, \ldots, V_n$. At each i-th level, we arrange the sets in lexicographic order.

The third stage. Consider a cube E_n^2 of sets of lengths n (Figure 1) in lexicographic order

$$\left(r_2 = 2\,]\log_2 m[\, + \,]\sqrt{\log_2 m}\,[\right)$$

$$(r_1 =]\log_2 m[+ 1)$$

It is clear that according to theorem 2.2 on sets $\tilde{\alpha}$ of levels V_i, $i < r_1$, $g(\tilde{\alpha}) = 0$.

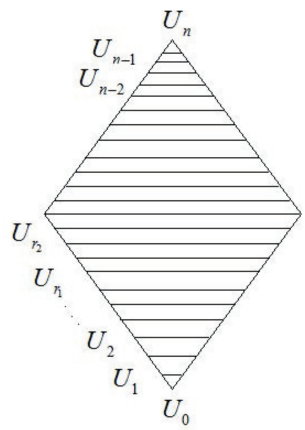

FIGURE 1. Lexicographic order of cube E_n^2

According to theorem 2.2, almost always on sets $\tilde{\alpha}$ of levels V_j, $j < r_2$, $g(\tilde{\alpha}) = 1$. Therefore, at this stage, we construct a modified N.N. Katerinochkina [9–11] algorithm of search for m.l.u. of function $g(\tilde{y})$ in the class of sets of cube E_n^2 of levels $V_{r_2}, V_{r_{2-1}}, \ldots, V_{r_1}$.

Algorithm A_1 consists of two blocks.

The first block. The first block. First, the algorithm calculates the value of g at the leftmost set of the r_1-th level. Let the value of g at the i-th step be calculated at some set $\tilde{\alpha}_i$, $r_{2-i}+1$-th level . If $g(\alpha_i) = 1$, then on the $i+1$-th step we calculate the value of on the leftmost set of the α_{i+1}, r_{2-i}-th level. If $g(\alpha_i) = 0$, then we proceed to the second block of the algorithm. If $g(\alpha_{r_2-r_1+1}) = 1$ is obtained at the $r_2 - r_1 + 1-$th step, then m.l.u. is $\alpha_{r_2-r_1+1}$ and the algorithm ends its work.

The second block. Let the $\tau+1$ steps $(0 \leq \tau \leq r_2 - r_1)$, be taken by the beginning of the second block of the algorithm, that is, the values of $g(\tilde{\alpha}_1) = g(\alpha_2) = \ldots = g(\tilde{\alpha}_\tau) = 1$ and $g(\tilde{\alpha}_{\tau+1}) = 0$, are calculated where $\tilde{\alpha}_1, \tilde{\alpha}_2, \ldots, \tilde{\alpha}_n$ are the leftmost sets of levels $V_{r_2}, V_{r_{2-1}}, \ldots, V_\tau$, respectively. Then, at the $(\tau+2)-$th step, we calculate the value of g on the set $\tilde{\alpha}_{\tau+2}$, immediately following $\tilde{\alpha}_{\tau+1}$ in the lexicographic order at the $r_2 - \tau-$th level. Let the value of g on some set $\tilde{\alpha}$, j-th level, where $j \geq r_2 - \tau$, be calculated at $i-$th step $i \geq \tau + 2$. If $g(\tilde{\alpha}_i) - 0$, then, at the $i+1$-th step, we calculate the value of on the set $\tilde{\alpha}_{i+1}$ immediately following $\tilde{\alpha}_i$ of the same j-th level. In the case $g(\tilde{\alpha}_i) = 1$, at the $i+1-$th step, we calculate the value of g on the set $\tilde{\alpha}_{i+1}$ of the $j-1$-th level, which immediately follows the set of the $j-1$-th level, obtained from $\tilde{\alpha}_i$ by zeroing the rightmost unit bit (since in all sets to the left $g(\beta) = 0$).

If no such set exists, then the process stops. Let the algorithm take r steps to stop. We get a chain of sets $\tilde{\alpha}_1, \tilde{\alpha}_2, \ldots, \tilde{\alpha}_r$ on which the value of g was subsequently calculated. Then m.l.u. is a set $\tilde{\alpha}_S$ from this chain with the maximum number S such that $f(\alpha_S) = 1$ [12, 13].

If there is no set $\tilde{\alpha}$ that $g(\tilde{\alpha}) = 1$, on the class of sets of levels V_{r_2}, then proceed to the fourth stage of the algorithm.

The fourth stage. At this stage, we apply the algorithm A_1 for searching m.l.u. of function g in a class of sets of levels

$$V_{r_2+1}, V_{r_2+2}, \ldots, V_{2(r_2+1)-r_1}.$$

If there is m.l.u. in the class of sets of these levels, then the algorithm ends its work. If not, proceed to the fifth stage.

The fifth stage. Let $t = 2(r_2 + 1) - r_1$. This stage consists of the following blocks.

The first block. At the first step of this block, we calculate the value of g at an arbitrary set α_1 $t+1$-th level. Let the value of g at a certain set $\tilde{\alpha}_i$ $t+1$-th level $(1 \leq i \leq n-t)$ be calculated at the $i-$th step. If $g(\tilde{\alpha}_i) = 0$, then at the $i+1$-th step we calculate the value of g on the set $\tilde{\alpha}_{i+1} \in V_{t+i+1}$, obtained from α_i by replacing the right-most zero coordinate with a unity. If, $g(\alpha_i) = 1$, then, proceed to the next block.

The second block. Let the τ steps and $P = t + \tau$ be completed by the beginning of the second block. We apply the algorithm A_1 to search for m.l.u. of function g in the class of sets $V_p, V_{p-1}, \ldots, V_{t+1}$.

3. Description of the MTT program.

Consider the MTT program, the functional block diagram of which is given in the following form (Figure 2):

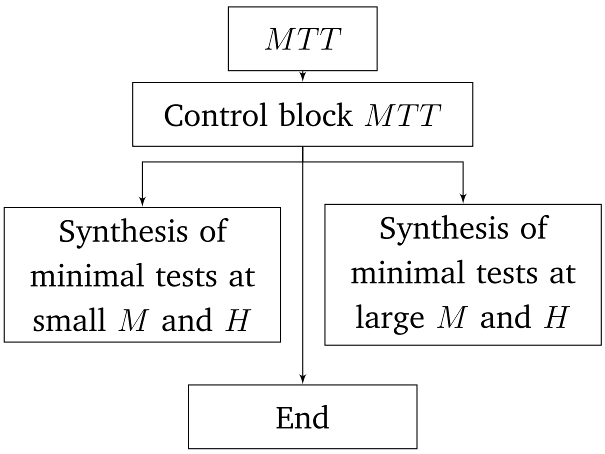

Figure 2. Program block diagram

The program is designed to build dead-end and minimal tests of an arbitrary binary table. It consists of a control block and two fundamental blocks (Figure 2).

In the control block, during the analysis of the in-put information, preparatory work is performed and control is transferred, depending on the input information, to the first or second block. The functions of the first and second blocks are almost identical. The only difference is that the first block works as a binary table, and the second - as a coded one, the decoding of which forms a binary table.

In its operation, the MTT program uses the following procedures.

Prosedure $A1$. The access to it is as follows: $A1\,(M, B)$ where M - is the number of output quantities, B - is the array of output quantities.

Outputs. To display M numbers from a one-dimensional array B.

Assignment. To display natural numbers by one space on the left and two spaces on the right.

Application. To display the table significance by rows and columns of an arbitrary binary table.

Prosedure $A2$. The access is $A2\,(M, N, T, ST)$, where:

(i) M is the number of rows in the table;
(ii) N is the number of columns in the table;
(iii) $T\,[I : M, I : N]$ is the table;
(iv) ST is the output format.

Outputs. Output of table T.

Assignment. To display the table in the specified format ST so that each row is displayed as a new paragraph.

Application. To display the input and output and intermediate tables when constructing a minimal test of the input table.

Procedure a function $A3$. Access: $A3\,(K,\ \ N,\ \ B)$, where K is the length of the binary set, NB is the array of the binary set.

Outputs. Decimal code of the set $A3$.

Assignment. To calculate the decimal code of an arbitrary set.

Algorithm. The decimal code of the set NB is calculated by the formula

$$A3 = \sum_{i=1}^{k} NB_i 2^{k-i}.$$

Application. For calculating table significance by rows and columns when constructing a minimal table test.

Procedure-function $A4$. Access:

$$A4(LP, K, N),$$

where $LP-$is the feature of selecting or deleting (if $LP = 0$ the $K-$th binary bit is deleted, if not, the $K-$th binary bit of the natural number N is selected); $K-$is the bit number; $N-$is a natural number.

The output information $A4$ - is a selected binary bit or a natural number obtained after deleting the specified binary bit.

The algorithm.

(i) The integer and fractional parts of the division N are selected:

$$\alpha_S = \left[N|\, 2^{k-1}\right], \;\; \alpha_D = N - \alpha_S 2^{k-1}.$$

(ii) The whole part of the division is selected α_S:

$$\alpha\alpha_S = [\alpha_S/2\,].$$

(iii) $\beta = \alpha\alpha_S 2^{k-1} + \alpha_D$ is calculated.

(iv) $\gamma = \left(N - \alpha\alpha_S 2^k - \alpha_D\right)/2^{k-1}$ is calculated.

If $LP = 0$, $\beta-$ the result of deleting the K-th binary bit of the number N is selected; in the case $LP = 1$, $\gamma-$the result of selecting the K-th binary bit of the number N is selected.

Application. To search for the minimal test of an arbitrary binary table.

Assignment. To select and delete a binary bit of a natural number.

Procedure $A5$. Access:

$$A5\,(LP, M, N, K, T)\,,$$

where $LP-$is the feature of deleting a row or column (at $LP = 0$, a row is deleted; at $LP = 1$, a column is deleted, in other cases, no deletion is performed); $T-$is the table; $M-$ is the number of rows in table $T-$; $N-$ is the number of columns in table T; $K-$ is the number of rows or columns to be deleted.

Output information. Table T.

Assignment. To delete (exclude) some row or column of the table.

Application. To search for the maximal test of an arbitrary binary table.

Procedure $A6$. Access:

$$A6\,(LP, N, K, T, Z)\,,$$

where $LP-$is the feature of selecting a row or a column, $LP \in \{0,1\}$ (at $LP = 0$ a row is selected; at $LP = 1$, a column is selected; in other cases, no selection is made); $-$ is the number of rows and $N-$ is the number of columns in table T; $K-$ is the number of the selected row or column; T is the table; $Z-$is the vector-result.

Outputs. Vector Z.

Assignment. To select a row or column of an arbitrary table.

Application. To calculate the table significance by rows and tables.

Procedure function $\boldsymbol{NR}$. Access:

$$NR\,(N, M, Z)\,,$$

where $M, \quad N-$ are the natural numbers $(M \leq N)\,;\, Z-$is the original set.

Outputs. The set Z and NR, are obtained at $NR = 0$; is the final set.

Assignment. To construct sequentially a class of sets in lexicographic order from N numbers by M pieces.

Application. To search for the minimal test of a table.

4. Instructions for using the MTT program.

The program is designed in the form of a procedure, the access to which has the form

$$MTT\,(N, M, T, LP, BN, BM)$$

where $N-$ is the number of columns, $M-$ is the number of rows, $T-$ is the table, $LP-$is the feature of finding the first minimal test or the first N minimal tests (if there are so many) of the table; BN and $BM-$ are the arrays $(BN\,[1:N]\,, BM\,[1:M, 1:N])$ where the results are placed.

Input information is:

(i) $T-$table

(ii) $N-$number of columns

(iii) number of rows and $LP-$feature (if $LP = 1$, the first minimal test of the table is found, if not - the first minimal tests). Output results are placed in arrays BN and BM.

In the $i-$th cell of the array BN, there is a decimal code of the columns that form the test.

Decoding the elements of array BM yields N sub-tables T that form the test.

5. Test case

Table 2 is taken as an example, consisting of twelve rows and ten examples of columns, each of them.

TABLE 2. Test table

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	1
0	0	0	0	0	1	1	0	0	0	1	0	1	1	1	0	0	0	0	1
0	1	1	1	1	0	1	0	0	1	0	1	0	0	1	0	0	0	1	0
1	0	0	0	1	1	1	0	0	1	0	1	0	1	0	0	0	1	0	1
0	0	0	1	0	0	0	0	1	1	1	1	1	1	1	0	0	0	1	1
0	1	1	1	0	1	1	1	1	1	0	1	0	0	0	0	1	0	1	0
0	0	1	0	0	0	1	1	1	0	1	0	0	1	1	1	0	1	1	1
1	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	1	0
0	0	1	1	0	0	1	1	1	0	1	0	0	1	1	1	0	1	1	1
1	1	1	1	0	1	0	0	0	0	1	0	0	0	1	0	1	0	1	1

Columns numbered in the table 2 - 4, 13, 17, 19, 20 were obtained as the first minimal test.

Conclusion

The article developed algorithms for constructing minimum (dead-end) tests, testers, when solving problems of recognition and forecasting based on methods of decoding and finding the maximum upper zero of monotone Boolean functions. To achieve the set goals, methods of solving problems of decoding and finding the maximum upper zero of monotone Boolean functions are investigated, an approximate parametric algorithm for solving these problems is constructed.

References

[1] V. E. Kuznetsov: *On one stochastic algorithm for calculating information characteristics of tables using the test method*, Discrete Analysis, Novosibirsk, Computing Center SB AS USSR, **23** (1973), 8–23.

[2] N. N. Katerinochkina: *Searching for the maximum upper zero of a monotone function of Boolean algebra*, Nauka, Reports of AS USSR, **224**(3) (1975), 557–560.

[3] E. V. Dyukova: *On an asymptotically optimal algorithm for constructing irredundant tests*, Reports of AS USSR, **233**(4) (1977), 527–530.

[4] A. V. Kabulov I. H. Normatov I. I. Kalandarov, A. A. Karimov: *Algorithmic Method of organization of Specialized Workshops*, Inter/Jour.IJARSET, **5**(4) (2018), 5670 –5675.

[5] I. H. Normatov: *Principle of independence of continuation of functions multivalued logic from coding*, Journal of Physics, (2018) 1210.

[6] A. V. Kabulov, E. Urunbaev, I. H. Normatov, A. O. Ashurov : *Logical method for constructing the optimal corrector of fuzzy heuristic algorithms* , 2019 International Conference on Information Science and Communications Technologies (ICISCT), (2019), 1–4.

[7] A. V. Kabulov, I. H. Normatov, A. O. Ashurov: *Computational methods of minimization of multiple functions*, Journal of Physics, (2019), 1260.

[8] A. V. Kabulov, I. H. Normatov : *About problems of decoding and searching for the maximum upper zero of discrete monotone functions*, Journal of Physics Conference Series, **6** (2019), 1260.

[9] A. V. Kabulov, E. Urunbaev, I. H. Normatov, A. O. Ashurov : *Synthesis methods of optimal discrete corrective functions*, Advances in Mathematics: Scientific Journal, **9**(9) (2020), 6467–6482.

[10] A. V. Kabulov, I. H. Normatov, Sh. Boltaev, I. Saymanov : *Logic method of classification of objects with non-joining classesa*, Advancesin Mathematics: Scientific Journal, **9**(10) (2020), 1857–8365.

[11] I. H. Normatov, E. Kamolov: *Development of an algorithm for optimizing the technological process of kaolin enrichment*, IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), Vancouver, BC, Canada, (2020), 1–4.

[12] A. V. Kabulov, I. H. Normatov, A. Seytov, A. Kudaybergenov: *Optimal Management of Water Resources in Large Main Canals with Cascade Pumping Stations*, IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), Vancouver, BC, Canada, (2020), 1–4.

[13] A. V. Kabulov, I. H. Normatov, A. Karimov: *Algorithmization control of complex systems based on functioning tables*, Journal of Physics Conference Series, **6** (2020), 1–9.

Department of Information Security
National University of Uzbekistan named after Mirzo Ulugbek
University street, 4, 100174 Tashkent, Uzbekistan
Email address: anvarkabulov@mail.ru

Department of Information Security
National University of Uzbekistan named after Mirzo Ulugbek
University street, 4, 100174 Tashkent, Uzbekistan
Email address: ibragim_normatov@mail.ru

Department of Mathematical Modeling and Complex Programming
Samarkand State University
University boulevard 15, 140104 Samarkand, Uzbekistan
Email address: urunbayeverkin@mail.ru

Department of Algorithmization and Mathematical Modeling
Tashkent University of Information Technologies named after Muhammad al-Khwarizmi
Amir Temur street, 108, 100200 Tashkent, Uzbekistan
Email address: a.o.ashurov@gmail.com