This paper reports a research that established the possibility of increasing the effectiveness of the method of figurative transformations to minimize partially defined Boolean functions. The method makes it possible, without loss of functionality, to reduce the complexity of the minimization procedure, compared to sorting out binary definitions of partially defined Boolean functions. The interpretation of the result is that the 2-(n, b)-design, 2-(n, x/b)-design systems are a reflection of logical operations. Therefore, the identification of such combinatorial systems in the truth table of logical functions directly and unambiguously establishes the location of logical operations for equivalent transformations of Boolean expressions. This, in turn, implicates an algorithm for simplifying Boolean functions, including partially defined Boolean functions. Thus, the method of figurative transformations simplifies and speeds up the procedure for minimizing partially defined Boolean functions, compared to analogs. This indicates that the visual-matrix form of the analytical method still has the prospect of increasing its hardware capabilities, including in terms of minimizing partially defined Boolean functions.

It has been experimentally confirmed that the method of figurative transformations increases the efficiency of minimizing partially defined Boolean functions, compared with analogs, by 100–200 %.

There is reason to argue about the possibility of increasing the efficiency of minimizing partially defined Boolean functions in the main and polynomial bases by the specified method. The effectiveness of the method, in particular, is ensured by carrying out all operations of generalized gluing of variables for dead-end disjunctive normal forms (DNF), followed by the use of implicant tables; optimal combination of a sequence of logical operations for gluing variables

Keywords: minimization of partially defined Boolean functions by the method of figurative transformations, location of equivalent transformations

UDC 519.718
DOI: 10.15587/1729-4061.2023.273293

IMPLEMENTING THE METHOD OF FIGURATIVE TRANSFORMATIONS TO MINIMIZE PARTIALLY DEFINED BOOLEAN FUNCTIONS

Mykhailo Solomko
Corresponding author
PhD, Associate Professor
Department of Computer Engineering
National University of Water and Environmental Engineering
Soborna str., 11, Rivne, Ukraine, 33028
E-mail: doctrinas@ukr.net
Mykola Antoniuk
PhD, Associate Professor*
Ihor Voitovych
Doctor of Pedagogical Sciences, Professor*
Yuliia Ulianovska
PhD, Associate Professor
Department of Computer Science and Software Engineering
University of Customs and Finance
Vernadskoho str., 2/4, Dnipro, Ukraine, 49000
Nataliia Pavlova
PhD, Associate Professor**
Viacheslav Biletskyi
PhD, Associate Professor**
*Department of Information and Communication Technologies and Methods of Teaching Informatics***
Department of Information and Communication Technologies and Methods of Teaching Computer Science*
***Rivne State University of Humanities
S. Bandery str., 12, Rivne, Ukraine, 33028

Received date 11.11.2022
Accepted date 24.01.2023
Published date 28.02.2023

How to Cite: *Solomko, M., Antoniuk, M., Voitovych, I., Ulianovska, Y., Pavlova, N., Biletskyi, V. (2023). Implementing the method of figurative transformations to minimize partially defined Boolean functions. Eastern-European Journal of Enterprise Technologies, 1 (4 (121)), 6–25. doi: https://doi.org/10.15587/1729-4061.2023.273293*

1. Introduction

When making digital devices, situations arise when the Boolean function is not defined on all sets of variables. To overcome this type of uncertainty when creating an analytical model of a digital device, the procedure for identifying a function on indefinite sets is used. This redefinition is a powerful degree of freedom and a resource for qualitative optimization of partially defined Boolean functions.

There are two ways to redefine a function:

– $f_1(x_1, x_2, \ldots x_n)$ on all undefined sets, the output function $f(x_1, x_2, \ldots x_n)$ is redefined by unities;

– $f_0(x_1, x_2, \ldots x_n)$ on all undefined sets, the output function $f(x_1, x_2, \ldots x_n)$ is redefined by zeros.

Then Quine theorem, at the moment, derives the following statement: the minimum disjunctive normal form (DNF) of a non-fully defined Boolean function is defined as the disjunction of the shortest implicant of the function. These implicants together cover all the minterms of the perfect disjunctive normal form (PDNF) of the function, and among the selected simple implicants there are no redundant [1].

Boolean functions are a special case of not fully defined functions. Non-fully defined Boolean functions can be represented using fully defined Boolean functions, which are

obtained using appropriate redefinition (sorting through all possible substitutions 0 or 1 instead of «–» *don't care* – indifferent or indefinite value) [2]. Then the procedure for minimizing a partially defined Boolean function should be carried out on each defined function and the optimal result should be selected.

With an increase in the number of indefinite sets, the set of fully defined Boolean functions increases significantly. For seven, for example, indefinite sets of variables, there are $2^7=128$ different ways of binary redefinition of partially defined Boolean functions, and, therefore, the complexity of the function minimization procedure will increase.

Thus, a relevant aspect of theoretical research into minimizing partially defined Boolean functions by the method of figurative transformations is:

– to reduce the number of techniques for binary redefinition of partially defined Boolean functions;

– to identify opportunities to improve the procedure for minimizing and expanding the apparatus for the synthesis of digital components based on partially defined Boolean functions for their use in digital technologies.

Particularly relevant are theoretical studies on minimizing partially defined Boolean functions aimed at improving such factors as:

– visual-matrix methods of minimizing partially defined Boolean functions of the main and polynomial basis;

– the cost of technology to minimize partially defined Boolean functions;

– ensuring the reliability of the obtained result of minimizing partially defined Boolean functions.

2. Literature review and problem statement

A new approach to minimizing Boolean expressions is considered in [3]. Despite the fact that the proposed method is general, attention is paid to exclusion functions or ESOP. A procedure has been developed that converts the problem from the region of Boolean algebra to the classical algebraic region. The resulting problem becomes a nonlinear, integer program, and to solve it, an original technique of branching and connections with several relaxations has been developed. The proposed procedure is especially suitable for minimizing incompletely defined Boolean functions, which is a complex problem in the Boolean domain. Numerical examples are given to demonstrate the feasibility of the approach and performance, and possible future directions are described. The resulting nonlinear problems can sometimes be very complex, but their complex solutions can solve open ESOP problems.

The algorithm for minimizing Boolean functions that have a small part of the defined sets of variables is discussed in [4]. Unlike other well-known minimization algorithms, the developed algorithm uses the strategy of «start from a large one», gradually reducing the value of the term until a simple implicant is generated. This approach allows for a very quick solution to the problem, even for examples with several hundred input variables and several hundred minterms with defined initial values. The software version of the algorithm provides better results (according to the criterion of execution time and minimization of the original function) compared to modern ESPRESSO. As with most heuristic and iterative algorithms, it is impossible to estimate the time complexity of completing an algorithm. In this regard, work [4] reports the average time that is required to perform one pass of the considered algorithm for different sizes of the input truth table.

The method of minimizing Boolean functions, which is based on nonlinear mixed integer programming, is presented in [5]. Experimental results show that the method produces the same or better results compared to other methods available in the literature. However, other methods do not guarantee a minimum solution. The main advantages of the proposed method of minimization are that the presented method guarantees obtaining a minimum function and can also be used to minimize incompletely defined Boolean functions.

To confirm the theoretical calculations of the method reported in [5], it is advisable to provide a demonstration example of simplifying a partially defined Boolean function by at least 4 variables.

Work [5] also states that all experimental examples were launched on a server with free access NEOS, which implements deterministic algorithms. However, the NEOS free access service limits the maximum calculation time to 8 hours, which is not enough to fully complete some of the examples. Therefore, for such examples, no definitive solution has been found, but instead the best has been found.

In programs where ROBDD has a great impact on the quality of the result (e.g., logical synthesis for FPGA implementations), there is a significant need for methods to minimize ROBDD dimensionality for incompletely defined functions. Minimization of ROBDD dimensionality for incompletely defined logical functions is discussed in [6]. The minimization method uses the symmetry properties of the ROBDD structure. However, the resulting ROBDD dimensionality is highly dependent on the order of the variable. The decisive point and problem are to position symmetric variables side by side and consider them as a fixed block. In this regard, there is a need to determine the necessary variable for the specified procedure. However, symmetric groups for partially defined Boolean functions are not fixed unambiguously. Therefore, there is a problem in determining the optimal division of variables into symmetric groups. There are two difficulties for dividing input variables into symmetric groups:

– first, how to find large sets of candidates of variables for symmetric groups (it must be borne in mind that it is impossible to check each subset of variables whether it is a symmetric group);

– secondly, how to combine symmetric groups in separated input variables for the case of simplification of partially defined Boolean functions.

Work [6] ends with experimental results confirming the effectiveness of the presented method. The procedure improves the symmetric dimensions of ROBDD by 51 % and, in combination with a slightly modified version, by 70 %.

Generalized rules for simplifying conjuncterms in a polynomial theoretical-set format are discussed in [7]. These rules are based on the proposed theorems for different initial conditions for the transformation of paired conjuncterms, the Hemming distance between which can be arbitrary. These rules can be used to minimize arbitrary logical functions (including partially defined Boolean functions) with n variables in a polynomial theoretical-set format. The advantages of the proposed rules for simplifying functions are illustrated with examples.

The vast majority of functions and their systems minimized by the proposed method in [7] showed the best results. This is due to the fact that conjuncterms with a Hemming distance $d\geq 3$ are involved in the transformation process. Note that the theory is focused on conclusions about the

inexpediency of further minimization in the polynomial format of Boolean functions if the distance between an arbitrary pair of its conjuncterms was $d \geq 3$. When applying the theorems [7] to the function f, which is given not by two, but by a larger number of conjuncterms of different ranks, between arbitrary pairs for which there is a different distance d, it is quite possible to further simplify it. This is explained by the fact that the set of transformed PTMFs $Y^{\oplus}$, with which the selected pair with a distance of $d \geq 3$ is replaced, may contain elements that, with other elements of the given function f, will form pairs with a distance of $d < 3$. Moreover, despite the increase in the power of the newly formed set, after applying the corresponding rules of theorems [7] to pairs with small d, it is possible to obtain a minimal PTMF $Y^{\oplus}$. As a result, the chance of effectively simplifying the set of conjuncterms increases. This, in turn, makes it possible to reduce the cost of implementing the minimized function.

The search procedure for such elements is combinatorial in nature. After each replacement of a selected pair of conjuncterms of a given PTMF $Y^{\oplus}$ by a certain set of formed PTMFs $Y^{\oplus}$, a new set is obtained in which the distance d between the new pairs must be determined. Having selected from them the elements with the minimum d, it is necessary to apply the rules of the corresponding theorem and build a new set again, etc.

Synthesis from partial specifications of logical circuits (LSFPS) is reported in [8]. LSFPS is a problem of finding a hardware implementation of partially defined Boolean functions. Logical synthesis from partial specifications (LSFPS) introduces the additional concept of «*don't know*» to terms that are not in the specifications. The exact solution of LSFPS is a logical scheme of the optimal size of the corresponding problem, in which indefinite sets of variables are invalid. In practice, specification information may not be sufficient to determine the exact functionality, then the goal is to maximize the accuracy of the scheme over an available subset of indefinite sets. Therefore, to the traditional goal of minimizing the size of the scheme, the goal of maximizing the accuracy of the scheme when evaluated by a subset of indefinite sets is added. The problem is relevant because effective solutions can lead to hardware-friendly machine learning models that do not rely on black box approaches. LSFPS directly corresponds to the problem of automatic generation of optimal topologies for binary neural networks. In addition, the combination of an accurate solution with modern methods of logical synthesis will unlock unprecedented optimization capabilities. Previous work has proven the effectiveness of *approximate logical synthesis* (ALS) for the design of circuits with their sufficient accuracy. Nevertheless, these methods sacrifice the accuracy of the specifications, which excludes them from legitimate candidates for LSFPS. Paper [8] proposes the *restoration of accuracy*, which consists in the procedure for comparing the approximate version of the scheme with the new one, which satisfies the exact functionality of the specifications. Experimental testing showed a decrease in the number of gates by 17.38 % and a decrease in the depth of the logic scheme by 12.02 %. Using the procedure to restore the accuracy of the scheme based on decomposition gives an accuracy of 95.73 %, which exceeds the current ALS level at which the accuracy is 92.76 %.

Taking into account the extensive research into logical synthesis for high-performance systems, it is necessary to investigate its potential role in the development of machine learning techniques with hardware in mind. It is worth noting that some tasks of machine learning allow for formulation as a fundamental problem of logical synthesis.

The problem of finding an approximate Boolean scheme from a set of examples is considered in [9]. Many computer programs are inherently error-resistant. This reduces the accuracy of calculations in order to achieve greater efficiency for chip area, computational performance, and/or power consumption. In recent years, a number of automated methods have been proposed for approximate calculations; however, most of these methods require full knowledge of the exact or «golden» description of the scheme. At the same time, there is a significant interest in the synthesis of calculations based on examples, the form of training under management. Paper [9] presents the relationship between the controlled learning of Boolean schemes and the existing work on the synthesis of incompletely defined Boolean functions. It has been demonstrated that when viewed through the prism of machine learning, the latter work provides good learning accuracy but low test accuracy. The article compares with previous work from the 1990s, which uses reciprocal information to guide the search process, striving for a good generalization. By combining this early work with the current approach to logical function learning, a scalable and efficient machine learning approach for Boolean schemes can be achieved in terms of area/latency/test error compromise. The results of this study indicate that the proposed technique has the potential to create Boolean schemes with high accuracy from large training sets of examples with a large number of primary input data. However, this method is limited to only 1-biton output. New research is needed that will take into account word-level search that will help to effectively search for logical meanings of the scheme with multi-bit outputs.

Methods of simplification of partially defined Boolean functions, which are considered in sources [3–9], mainly use theoretical objects of related theory, such as Hemming weights, ordered binary decision diagrams (ROBDD), the transformation of a problem from the domain of Boolean algebra into a classical algebraic region, methods for finding approximate Boolean schemes, large training sets of examples with a large number of primary input data, ways to restore the accuracy of a logical scheme, machine learning techniques, as well as software. A mandatory technological point for the implementation of these algorithms and methods is the need for automated calculations.

The method of figurative transformations is based on binary combinatorial systems with repeated 2-(n, b)-design, 2-$(n, x/b)$-design, which are part of the binary structures of truth tables and provide unambiguous detection of the location of equivalent transformations to simplify Boolean functions. The set place of equivalent transformations implicates the algorithm for minimizing Boolean functions. This, in turn, makes it possible to reduce the complexity of the simplification procedure without loss of functionality, compared with algorithms and methods for simplifying partially defined Boolean functions discussed in works [3–9]. By qualification, the method of figurative transformations belongs to the visual-matrix form of the analytical method [10] and does not exclude the manual technique of minimizing partially defined Boolean functions.

Thus, the algorithms and methods created by the software for them, which cover the general procedure for simplifying partially defined Boolean functions [3–9] and the method of figurative transformations, occupy different approaches (principles of minimization). Therefore, they imply different perspectives regarding the possibility of algorithmic minimization of partially defined Boolean functions.

Combinatorial systems of 2-(n, b)-design, 2-$(n, x/b)$-design are a representation of logical operations, the detection of which simplifies and speeds up the procedure for minimizing partially defined Boolean functions compared to analogs. This indicates that the visual-matrix form of the analytical method still has the prospect of increasing its hardware capabilities, including in terms of minimizing partially defined Boolean functions.

And this is reason to believe that the software and technological base, which is represented by algorithms and methods with theoretical objects of related theories, [3–9], is insufficient to conduct theoretical research on the optimal minimization of partially defined Boolean functions. This predetermines the need for research with equivalent figurative transformations to minimize partially defined Boolean functions.

In applied terms, the method of figurative transformations will provide an expansion of the possibilities of the technology for designing digital components based on partially defined Boolean functions in the main $\{\vee, \wedge, \neg\}$ and polynomial $\{\wedge, \oplus, 1\}$ bases.

3. The aim and objectives of the study

The aim of this work is to extend the method of figurative transformations to minimize disjunctive normal forms (DNF), conjunctive normal forms (CNF), and polynomial normal forms (PNF) of the partially defined Boolean functions in the Boolean and Reed-Muller bases. This will make it possible to simplify, increase the minimization performance of partially determined Boolean functions in the main and polynomic bases, using the algebraic apparatus of these bases.

To accomplish the aim, the following tasks have been set:

– to determine the optimal combination of the sequence of logical operations of gluing variables – simple and super-gluing in the initial truth table of a partially given Boolean function;

– to establish direct implication between the detection of locations of equivalent transformations by the combinatorial systems of 2-(n, b)-design, 2-$(n, x/b)$-design and the algorithm for minimizing Boolean functions;

– to analyze the result of simplification of a partially defined Boolean function by the method of figurative transformations and an example of minimization by the heuristic method in order to compare the cost of implementing the resulting minimum function;

– to conduct a comparative analysis of the results of simplification of partially defined Boolean functions of the main basis by the method of figurative transformations and minimization methods borrowed from other authors in order to compare the cost of implementing the minimum function and the number of procedural steps;

– to analyze the result of simplification of a partially defined Boolean function in the Reed-Muller basis by the method of figurative transformations and an example of minimizing partially defined Boolean functions by the method of decoupling conjuncterms in order to compare the cost of implementing the minimum function.

4. The study materials and methods

Minimization of dead-end DNF (DCNF), in particular, is possible by carrying out all operations of generalized gluing of DDNF variables (DCNF). The next step is to use implicant tables to detect unnecessary simple implicants and to select term functions with a minimum number of inversions.

Example 1. It is required, by the method of figurative transformations, to minimize the DNF of a partially defined Boolean function specified in binary form:

$$f(x_1, x_2, x_3, x_4)=(1--0010010-01--1). \quad (1)$$

Solution:

$$f_{\text{DeDNF}}(x_1,x_2,x_3,x_4)=$$

No.	x_1	x_2	x_3	x_4	f
0	0	0	0	0	1
5	0	1	0	1	1
8	1	0	0	0	1
12	1	1	0	0	1
15	1	1	1	1	1
1	0	0	0	1	–
2	0	0	1	0	–
10	1	0	1	0	–
13	1	1	0	1	–
14	1	1	1	0	–

$=$

x_1	x_2	x_3	x_4	f
	0		0	1
0		0	1	1
1	1			1

. (2)

The first matrix in expression (2) represents the truth table of the partially defined DNF of the function $f(x_1, x_2, x_3, x_4)$ (1). The last matrix in expression (2) represents the dead-end DNF of the function $f(x_1, x_2, x_3, x_4)$ (1) which takes the following form:

$$f_{\text{DeDNF}}(x_1,x_2,x_3,x_4)=x_1x_2+\overline{x_1}\,\overline{x_3}x_4+\overline{x_2}\,\overline{x_4}. \quad (3)$$

The dead-end DNF (3) contains four inversions. To reduce the number of inversions, we shall carry out all possible operations of generalized gluing of variables in DDNF (3) followed by the use of an implicant table (Table 1) to identify unnecessary simple implicants and select simple implicants with a minimum number of inversions:

$$f_{\text{DeDNF}}(x_1,x_2,x_3,x_4)=$$

x_1	x_2	x_3	x_4	f
	0		0	1
0		0	1	1
1	1			1

$=$

x_1	x_2	x_3	x_4
1			0
	0		0
0		0	1
0	0	0	
1	1		
	1	0	1

.

Table 1

Implicant table of the function f(x1, x2, x3, x4) DNF (1)

No.	x_1	x_2	x_3	x_4	f	1– –0	–0–0	0–01	000–	11– –	–101
0	0	0	0	0	1	–	•	–	•	–	–
5	0	1	0	1	1	–	–	•	–	–	•
8	1	0	0	0	1	•	•	–	–	–	–
12	1	1	0	0	1	•	–	–	–	•	–
15	1	1	1	1	1	–	–	–	–	•	–

Table 1 demonstrates that simple implicants 1– –0, 000– are redundant. Among the simple implicants 0–01 and –101 you need to choose the latter since it contains fewer inversions. Then the function (1) MDNF will take the following form:

$$f_{\text{MDNF}}(x_1,x_2,x_3,x_4)=x_1x_2+x_2\overline{x_3}x_4+\overline{x_2}\,\overline{x_4}. \quad (4)$$

The minimum DNF (4) contains three inversions, which is one less inversion compared to the dead-end DNF (3).

Identity in Boolean algebra is defined as the equality of two expressions, which holds on the entire set of values of variables.

The peculiarity of using identities for partially defined Boolean functions is that an arbitrary identity can hold for one partially defined Boolean function and not hold for another partially defined Boolean function.

For a partially defined Boolean function $f(x_1, x_2, x_3, x_4)$, given in canonical form:

$$f(x_1,x_2,x_3,x_4)= =\sum m(3,5,6,9,12,15)+\sum d(1,2,8,11), \quad (5)$$

which contains a total of 6 minterms and 4 indefinite sets of variables, the following two identities hold:

$$1.\ x_1\oplus x_3\oplus x_4+\overline{x_2}x_4=x_1\oplus x_3\oplus x_4\oplus\overline{x_2}x_4. \quad (6)$$

Verification of identity (6) on the defined sets of variables of function (5) is given in Table 2.

Table 2

Verification of identity (6) on the defined sets of function (5) variables

No.	x_1	x_2	x_3	x_4	$(x_1\oplus x_3\oplus x_4)+\overline{x_2}x_4$	f	$x_1\oplus x_3\oplus x_4\oplus\overline{x_2}x_4$	f
0	0	0	0	0	$(0_1\oplus 0_3\oplus 0_4)+\overline{0_2}0_4$	0	$0_1\oplus 0_3\oplus 0_4\oplus\overline{0_2}0_4$	0
1	0	0	0	1	–	–	–	–
2	0	0	1	0	–	–	–	–
3	0	0	1	1	$(0_1\oplus 1_3\oplus 1_4)+\overline{0_2}1_4$	1	$0_1\oplus 1_3\oplus 1_4\oplus\overline{0_2}1_4$	1
4	0	1	0	0	$(0_1\oplus 0_3\oplus 0_4)+\overline{1_2}0_4$	0	$0_1\oplus 0_3\oplus 0_4\oplus\overline{1_2}0_4$	0
5	0	1	0	1	$(0_1\oplus 0_3\oplus 1_4)+\overline{1_2}1_4$	1	$0_1\oplus 0_3\oplus 1_4\oplus\overline{1_2}1_4$	1
6	0	1	1	0	$(0_1\oplus 1_3\oplus 0_4)+\overline{1_2}0_4$	1	$0_1\oplus 1_3\oplus 0_4\oplus\overline{1_2}0_4$	1
7	0	1	1	1	$(0_1\oplus 1_3\oplus 1_4)+\overline{1_2}1_4$	0	$0_1\oplus 1_3\oplus 1_4\oplus\overline{1_2}1_4$	0
8	1	0	0	0	–	–	–	–
9	1	0	0	1	$(1_1\oplus 0_3\oplus 1_4)+\overline{0_2}1_4$	1	$1_1\oplus 0_3\oplus 1_4\oplus\overline{0_2}1_4$	1
10	1	0	1	0	$(1_1\oplus 1_3\oplus 0_4)+\overline{0_2}0_4$	0	$1_1\oplus 1_3\oplus 0_4\oplus\overline{0_2}0_4$	0
11	1	0	1	1	–	–	–	–
12	1	1	0	0	$(1_1\oplus 0_3\oplus 0_4)+\overline{1_2}0_4$	1	$1_1\oplus 0_3\oplus 0_4\oplus\overline{1_2}0_4$	1
13	1	1	0	1	$(1_1\oplus 0_3\oplus 1_4)+\overline{1_2}1_4$	0	$1_1\oplus 0_3\oplus 1_4\oplus\overline{1_2}1_4$	0
14	1	1	1	0	$(1_1\oplus 1_3\oplus 0_4)+\overline{1_2}0_4$	0	$1_1\oplus 1_3\oplus 0_4\oplus\overline{1_2}0_4$	0
15	1	1	1	1	$(1_1\oplus 1_3\oplus 1_4)+\overline{1_2}1_4$	1	$1_1\oplus 1_3\oplus 1_4\oplus\overline{1_2}1_4$	1

Table 2 demonstrates that the left and right parts of identity (6) on the defined sets of variables of function (5) take the same value, so identity (6) holds for a partially defined function (5).

$$2.\ x_1\oplus x_3\oplus x_4\oplus\overline{x_2}x_4(x_1\oplus x_3)=x_1\oplus x_3\oplus x_4\oplus\overline{x_2}x_4. \quad (7)$$

Verification of identity (7) on the defined sets of variables of function (5) is given in Table 3.

Table 3

Verification of identity (7) on the defined sets of function (5) variables

No.	x_1	x_2	x_3	x_4	$x_1\oplus x_3\oplus x_4\oplus\overline{x_2}x_4(x_1\oplus x_3)$	f	$x_1\oplus x_3\oplus x_4\oplus\overline{x_2}x_4$	f
0	0	0	0	0	$0_1\oplus 0_3\oplus 0_4\oplus\overline{0_2}0_4(0_1\oplus 0_3)$	0	$0_1\oplus 0_3\oplus 0_4\oplus\overline{0_2}0_4$	0
1	0	0	0	1	–	–	–	–
2	0	0	1	0	–	–	–	–
3	0	0	1	1	$0_1\oplus 1_3\oplus 1_4\oplus\overline{0_2}1_4(0_1\oplus 1_3)$	1	$0_1\oplus 1_3\oplus 1_4\oplus\overline{0_2}1_4$	1
4	0	1	0	0	$0_1\oplus 0_3\oplus 0_4\oplus\overline{1_2}0_4(0_1\oplus 0_3)$	0	$0_1\oplus 0_3\oplus 0_4\oplus\overline{1_2}0_4$	0
5	0	1	0	1	$0_1\oplus 0_3\oplus 1_4\oplus\overline{1_2}1_4(0_1\oplus 0_3)$	1	$0_1\oplus 0_3\oplus 1_4\oplus\overline{1_2}1_4$	1
6	0	1	1	0	$0_1\oplus 1_3\oplus 0_4\oplus\overline{1_2}0_4(0_1\oplus 1_3)$	1	$0_1\oplus 1_3\oplus 0_4\oplus\overline{1_2}0_4$	1
7	0	1	1	1	$0_1\oplus 1_3\oplus 1_4\oplus\overline{1_2}1_4(0_1\oplus 1_3)$	0	$0_1\oplus 1_3\oplus 1_4\oplus\overline{1_2}1_4$	0
8	1	0	0	0	–	–	–	–
9	1	0	0	1	$1_1\oplus 0_3\oplus 1_4\oplus\overline{0_2}1_4(1_1\oplus 0_3)$	1	$1_1\oplus 0_3\oplus 1_4\oplus\overline{0_2}1_4$	1
10	1	0	1	0	$1_1\oplus 1_3\oplus 0_4\oplus\overline{0_2}0_4(1_1\oplus 1_3)$	0	$1_1\oplus 1_3\oplus 0_4\oplus\overline{0_2}0_4$	0
11	1	0	1	1	–	–	–	–
12	1	1	0	0	$1_1\oplus 0_3\oplus 0_4\oplus\overline{1_2}0_4(1_1\oplus 0_3)$	1	$1_1\oplus 0_3\oplus 0_4\oplus\overline{1_2}0_4$	1
13	1	1	0	1	$1_1\oplus 0_3\oplus 1_4\oplus\overline{1_2}1_4(1_1\oplus 0_3)$	0	$1_1\oplus 0_3\oplus 1_4\oplus\overline{1_2}1_4$	0
14	1	1	1	0	$1_1\oplus 1_3\oplus 0_4\oplus\overline{1_2}0_4(1_1\oplus 1_3)$	0	$1_1\oplus 1_3\oplus 0_4\oplus\overline{1_2}0_4$	0
15	1	1	1	1	$1_1\oplus 1_3\oplus 1_4\oplus\overline{1_2}1_4(1_1\oplus 1_3)$	1	$1_1\oplus 1_3\oplus 1_4\oplus\overline{1_2}1_4$	1

Table 3 demonstrates that the left and right parts of identity (7) on the defined sets of variables of function (5) take the same value, so identity (7) holds for a partially defined function (5).

5. Results of minimization of partially defined Boolean functions by the method of figurative transformations

5. 1. Optimal combination of a sequence of different methods of logical operations for gluing variables

The efficiency of minimizing partially defined Boolean functions (as well as fully defined) by the analytical method depends on combining the sequence of logical operations using different methods of gluing variables – simple and super-gluing in the first, and, in some cases, in the second binary matrix [11]. In [11] it is also demonstrated that the sequence of logical operations of super-gluing of variables and simple gluing of variables is not always optimal. For example, for the perfect disjunctive normal form (PDNF) of the function $f(x_1, x_2, x_3, x_4)$:

$$f(x_1,x_2,x_3,x_4)=\begin{vmatrix}0&0&1&1\\0&1&0&1\\1&0&0&0\\1&0&0&1\\1&0&1&1\\1&1&0&1\\1&1&1&0\\1&1&1&1\end{vmatrix}=\begin{vmatrix}&0&1&1\\&1&0&1\\1&0&0&\\1&1&1&\end{vmatrix}, \quad (8)$$

the optimal start of minimization is the logical operation of simple gluing of variables. Despite the fact that logical functions, like (8), do not happen often, in general, it is necessary to identify an optimal combination of a sequence of logical operations using different techniques of gluing variables – simple and super-gluing.

In the initial combination of a sequence of logical operations of super-gluing and simple gluing of variables, redundant simple implicants can be detected using an implicant table. This gives a choice in favor of the specified sequence of logical operations and provides unambiguous algorithm for minimizing Boolean functions. In particular, the operation of super-gluing variables for function (8) gives the following result:

$$f(x_1,x_2,x_3,x_4)= \begin{vmatrix} 0&0&1&1\\ 0&1&0&1\\ 1&0&0&0\\ 1&0&0&1\\ 1&0&1&1\\ 1&1&0&1\\ 1&1&1&0\\ 1&1&1&1 \end{vmatrix} = \begin{vmatrix} 0&0&1&1\\ 0&1&0&1\\ 1&0&0&0\\ 1&&&1\\ 1&1&1&0 \end{vmatrix} = \begin{vmatrix} &0&1&1\\ &1&0&1\\ 1&0&0&\\ 1&&&1\\ 1&1&1& \end{vmatrix}. \qquad (9)$$

Detection of redundant simple implicants for the result of simplification (9) is carried out using an implicant table (Table 4).

Table 4

Implicant table

Implicant / Implicant	0011	0101	1000	1001	1011	1101	1110	1111
–011	•	–	–	–	•	–	–	–
–101	–	•	–	–	–	•	–	–
1–1	–	–	–	•	•	•	–	•
100–	–	–	•	•	–	–	–	–
111–	–	–	–	–	–	–	•	•

Behold Table 4 'til comprehending that the simple implicant 1– –1 covers the most minterms of the PDNF of the original function – 1001, 1011, 1101, 1111, but is still redundant. Finally, we get the minimum function that coincides with the result of simplification (8):

$$f_{\text{MDNF}} = \begin{vmatrix} &0&1&1\\ &1&0&1\\ 1&0&0&\\ 1&&&1\\ 1&1&1& \end{vmatrix} = \begin{vmatrix} &0&1&1\\ &1&0&1\\ 1&0&0&\\ 1&1&1& \end{vmatrix}.$$

Thus, the initial combination of the sequence of logical operations of super-gluing of variables and simple gluing of variables in the first, and, in some cases, in the second binary matrix, with the possible use of an implicant table to detect unnecessary simple implicants, provides unambiguousness and sufficient efficiency of the algorithm for minimizing Boolean functions.

Another option to simplify expression (9) may be as follows:

$$f(x_1,x_2,x_3,x_4)= = \begin{vmatrix} &0&1&1\\ 1&0&0&\\ 1&&&1\\ 1&1&1& \end{vmatrix} = \begin{vmatrix} &0&1&1\\ &1&0&1\\ 1&0&0&\\ 1&0&0&1\\ 1&1&0&1\\ 1&0&1&1\\ 1&1&1&1\\ 1&1&1& \end{vmatrix} = \begin{vmatrix} &0&1&1\\ &1&0&1\\ 1&0&0&\\ 1&1&1& \end{vmatrix}. \qquad (10)$$

The result of simplification (10) coincides with the result of simplification (8) but is less obvious.

5. 2. Detection of locations of equivalent transformations using combinatorial systems of 2-(n, b)-design, 2-(n, x/b)-design

There are two types of uncertainty for the system in practice (Fig. 1): by input and output (either the input action cannot come from the outside, or the system's response to the input action is unimportant).

A partially defined Boolean function can be represented by a set of fully defined Boolean functions, which are obtained using the corresponding redefinition (sorting through all possible substitutions 0 or 1 instead of «–») (Fig. 2). Then the procedure for minimizing a partially defined Boolean function should be carried out on each defined function and the optimal result should be selected.

x_1	x_2	x_3	y
0	0	0	1
0	0	1	–
0	1	0	0
0	1	1	1
1	0	0	–
1	0	1	1
1	1	0	0
1	1	1	0

Uncertainties
Don't Care

Fig. 1. Uncertainties in the system

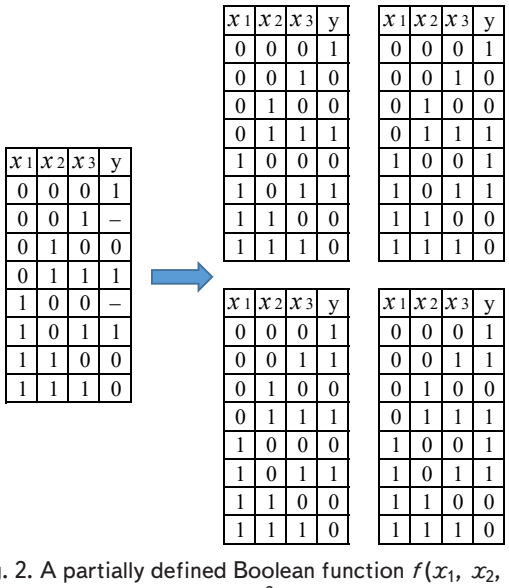

Fig. 2. A partially defined Boolean function $f(x_1, x_2, x_3)$ is represented by a set of $2^2=4$ fully defined Boolean functions $f(x_1, x_2, x_3)$

As the number of indefinite sets increases, the set of fully defined functions increases significantly. For six, for example, indefinite sets of variables, there are $2^6=64$ different ways of binary redefinition of partially defined Boolean functions, and, therefore, the complexity of the function minimization procedure will increase.

To reduce the number of techniques of binary redefinition of partially defined Boolean functions, it is necessary to choose a binary configuration that will provide conditions for super-gluing operations of variables and/or simple gluing of variables. It will also include defined sets of variables and some indefinite sets of variables, in particular. For a number of partially defined Boolean functions, their optimal minimization will require the use of all indefinite sets of variables.

Example 2. The partially defined Boolean function $f(A, B, C, D)$ is given by the truth table (Table 5). It is required to find the minimum DNF [12].

Table 5

Truth table of a partially defined Boolean function $f(A, B, C, D)$

No.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
x_1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
x_2	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
x_3	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
x_4	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
f	1	0	–	–	1	0	–	0	0	0	0	–	1	0	0	0

Solution:

$f_{\text{DeDNF}}(A,B,C,D)=$

	A	B	C	D	f
0	0	0	0	0	1
4	0	1	0	0	1
12	1	1	0	0	1
2	0	0	1	0	–
3	0	0	1	1	–
6	0	1	1	0	–
11	1	0	1	1	–

$=$

	A	B	C	D	f
0	0			0	1
12	1	1	0	0	1
3	0	0	1	1	–
11	1	0	1	1	–

$=$

	A	B	C	D	f
0	0			0	1
12		1	0	0	1
3	0	0	1	1	–
11	1	0	1	1	–

. (11)

The first matrix of expression (11) represents the truth table of the partially defined DNF of the function $f(A, B, C, D)$ (Table 5). Blocks 0, 4, 2, 6, highlighted in red, are subject to super-gluing the variables [13]. The result of logical operations of the first matrix is written to the second matrix of expression (11).

The minimal DNF takes the form:

$$f(A,B,C,D)=\overline{A}\,\overline{D}+B\overline{C}\,\overline{D}. \quad (12)$$

We note that the undefined sets of variables 3, 11 were not used during the simplification of the function $f(A, B, C, D)$ (Table 5) by the method of figurative transformations. This ultimately reduced the overall complexity of simplifying the function. The result of minimization (12) coincides with [12].

Combinatorial systems of 2-(n, b)-design, 2-$(n, x/b)$-design, which are part of the binary structures of truth tables, provide unambiguous detection of locations of equivalent transformations to simplify Boolean functions. The set place of equivalent transformations implicates the algorithm for minimizing Boolean functions. Thus, the 2-(n, b)-design, 2-$(n, x/b)$-design systems have an information capacity, which makes it possible to replace the binary definition of partially defined Boolean functions with an effect, such as in Fig. 2, with a method of figurative transformations. This, in turn, makes it possible to reduce the complexity of the minimization procedure without loss of functionality, compared to sorting out binary redefinitions of partially defined Boolean functions.

The location of equivalent transformations using the 2-(n, b)-design systems will be demonstrated by an example of minimizing the DNF of a partially defined Boolean function given by binary form [1]:

$$f(x_1, x_2, x_3, x_4)=(1--0-01--10-0-1-). \quad (13)$$

Solution:

Table 6

Truth table of the partially defined DNF of the function $f(x_1, x_2, x_3, x_4)$ (13)

No.	x_1	x_2	x_3	x_4	f
0	0	0	0	0	1
6	0	1	1	0	1
9	1	0	0	1	1
14	1	1	1	0	1
1	0	0	0	1	–
2	0	0	1	0	–
4	0	1	0	0	–
7	0	1	1	1	–
8	1	0	0	0	–
11	1	0	1	1	–
13	1	1	0	1	–
15	1	1	1	1	–

Blocks 0, 9, 1, 8 (highlighted in green) and blocks 6, 14, 7, 15 (highlighted in orange), each of which makes up a complete combinatorial system of 2 (2, 4)-design, are subject to the operation of super-gluing the variables [13]. The result of these logical operations is recorded in the following matrix (Fig. 3).

No.	x_1	x_2	x_3	x_4	f
–		0	0		1
–		1	1		1

Fig. 3. The result of minimizing the function $f(x_1, x_2, x_3, x_4)$ (13)

Since the undefined sets of variables 2, 4, 11, 13 of the function $f(x_1, x_2, x_3, x_4)$ (Table 6) do not represent the location of equivalent transformations, they are not used and are not displayed in the matrix in Fig. 3. This ultimately reduces the overall complexity of simplifying the DNF of the partially defined function (13). The minimum DNF of function (13) takes the following form:

$$f_{\text{MDNF}} = \overline{x_2}\,\overline{x_3} + x_2 x_3. \qquad (14)$$

The result of minimization (14) coincides with [1].

For eight indefinite sets of variables, there are $2^8=256$ different ways of binary redefinition of a partially defined Boolean function (13). Unlike the binary redefinitions of the example under consideration, only two combinatorial systems of 2-(2, 4)-design establish the location of the required equivalent transformations. This reduces the complexity of the minimization procedure, compared to sorting out the redefined functions, gives the result of simplification without losing the functionality of a given Boolean function and provides the efficiency of replacing binary definitions with the method of figurative transformations.

The location of equivalent transformations using a partially balanced combinatorial system of 2-(n, x/b)-design is demonstrated by the minimization of the CNF of the partially defined Boolean function (M^0, M^-) in Example 4.

5. 3. An exact and heuristic method for minimizing partially defined Boolean functions

Using the exact method of minimizing Boolean functions, it is possible to assess the quality of heuristic methods, and even determine the direction of creation of the heuristic method [14].

Example 3. It is required, by the method of figurative transformations, to simplify the DNF of a partially defined function $f(x_1, x_2, x_3, x_4, x_5, x_6)$, which is represented by the Carnot map [15].

d e f

		0		1	0		1
				0		1	
1		1	1	0	0		
					0		
	1			0	0	1	
0			0				
		0		1	1		0
	1	0	1			1	

a b c

Fig. 4. Partially defined Boolean function of six variables (Carnot map)

Solution:

$F(x_1, x_2, x_3, x_4, x_5, x_6) =$

No.	x_1	x_2	x_3	x_4	x_5	x_6	f
4	0	0	0	1	0	0	1
6	0	0	0	1	1	0	1
13	0	0	1	1	0	1	1
24	0	1	1	0	0	0	1
26	0	1	1	0	1	0	1
27	0	1	1	0	1	1	1
33	1	0	0	0	0	1	1
34	1	0	0	0	1	0	1
37	1	0	0	1	0	1	1
46	1	0	1	1	1	0	1
47	1	0	1	1	1	1	1
49	1	1	0	0	0	1	1
53	1	1	0	1	0	1	1
0	0	0	0	0	0	0	–
1	0	0	0	0	0	1	–
2	0	0	0	0	1	0	–
5	0	0	0	1	0	1	–
8	0	0	1	0	0	0	–
9	0	0	1	0	0	1	–
10	0	0	1	0	1	0	–
11	0	0	1	0	1	1	–
12	0	0	1	1	0	0	–
15	0	0	1	1	1	1	–
16	0	1	0	0	0	0	–
17	0	1	0	0	0	1	–
18	0	1	0	0	1	0	–

No.	x_1	x_2	x_3	x_4	x_5	x_6	f
19	0	1	0	0	1	1	–
20	0	1	0	1	0	0	–
21	0	1	0	1	0	1	–
22	0	1	0	1	1	0	–
25	0	1	1	0	0	1	–
28	0	1	1	1	0	0	–
29	0	1	1	1	0	1	–
32	1	0	0	0	0	0	–
36	1	0	0	1	0	0	–
38	1	0	0	1	1	0	–
39	1	0	0	1	1	1	–
40	1	0	1	0	0	0	–
41	1	0	1	0	0	1	–
42	1	0	1	0	1	0	–
45	1	0	1	1	0	1	–
48	1	1	0	0	0	0	–
50	1	1	0	0	1	0	–
51	1	1	0	0	1	1	–
52	1	1	0	1	0	0	–
57	1	1	1	0	0	1	–
59	1	1	1	0	1	1	–
60	1	1	1	1	0	0	–
61	1	1	1	1	0	1	–
62	1	1	1	1	1	0	–
63	1	1	1	1	1	1	–

=

No.	x_1	x_2	x_3	x_4	x_5	x_6	f
–		0	0			0	1
–					0	1	1
–	0	1		0	0	0	1
–	0	1		0	1	0	1
–	0	1		0	1	1	1
–	1		1	1	1		1

=

No.	x_1	x_2	x_3	x_4	x_5	x_6	f
–		0	0			0	1
–					0	1	1
–	0	1		0	0		1
–	0	1		0	1	0	1
–	0	1		0		1	1
–	1		1	1	1		1

=

No.	x_1	x_2	x_3	x_4	x_5	x_6	f
–		0	0			0	1
–					0	1	1
–	0	1		0	0		1
–	0	1		0	1	0	1
–	0	1		0		1	1
–	1		1	1	1		1
–	0	1		0	1		–

=

No.	x_1	x_2	x_3	x_4	x_5	x_6	f
–		0	0			0	1
–					0	1	1
–	0	1		0			1
–	0	1		0	1	0	1
–	0	1		0		1	1
–	1		1	1	1		1

=

No.	x_1	x_2	x_3	x_4	x_5	x_6	f
–		0	0			0	1
–					0	1	1
–	0	1		0			1
–	1		1	1	1		1

. (15)

The first matrix of expression (15) represents the truth table of the partially defined DNF of the function $f(x_1, x_2, x_3, x_4, x_5, x_6)$ (Fig. 4).

Blocks 13, 33, 37, 49, 53, 1, 5, 9, 17, 21, 25, 29, 41, 45, 57, 61, which are highlighted in red and make up the complete combinatorial system of 2-(4, 16)-design, and blocks 4, 6, 34, 0, 2, 32, 36, 38, which are highlighted in blue and make up the complete combinatorial system of 2-(3, 8)-design, as well as blocks 46, 47, 62, 63, which are highlighted in green and make up the complete combinatorial system of 2-(2, 4)-design, are subject to the operation of super-gluing the variables [13]. Blocks 24, 16 (highlighted in purple), and 26, 18 (highlighted in dark blue), as well as blocks 27, 19 (highlighted in brown), are subject to the operation of super-gluing the variables. The result of these logical operations is written to the second matrix of expression (15). In the second matrix of expression (15), undefined sets of variables are not displayed because they do not participate in the further simplification of the function.

As a result, the minimum function is obtained:

$$f_{\text{MDNF}} = \overline{x}_2\,\overline{x}_3\,\overline{x}_6 + \overline{x}_5 x_6 + \overline{x}_1 x_2 \overline{x}_4 + x_1 x_3 x_4 x_5, \qquad (16)$$

which contains two literals less than simplification using the heuristic method [15].

The cost of implementing the obtained minimum function (16) by the method of figurative transformations is $k_\theta/k_l/k_{in} = 4/12/6$, where k_θ, k_l, k_{in} – the number of conjuncterms, literals, and inverters, respectively.

The minimum function obtained by the heuristic method [15] is:

$$f_{\text{MDNF}} = \bar{e}f + \bar{b}\,\bar{c}\,\bar{f} + \bar{c}\,\bar{e} + \bar{a}cd + acde,$$

with an implementation cost $k_\theta/k_l/k_{in} = 5/14/8$.

5. 4. Minimization of partially defined Boolean functions of the main basis

Example 4. It is required to obtain the minimum orthogonal disjunctive normal form (MODNF) of the partially defined function $f(x_1, x_2, x_3, x_4, x_5)$, which is given by the matrices M^1 and M^0 [14], by the method of figurative transformations.

$M^1 =$

x_1	x_2	x_3	x_4	x_5	No.
0	0	1	0	1	1
0	1	1	0	0	2
1	0	0	1	0	3
0	0	1	1	1	4
1	0	0	1	1	5
1	0	1	1	0	6
1	1	0	1	0	7
1	1	1	0	0	8
1	1	1	1	0	9

, $M^0 =$

x_1	x_2	x_3	x_4	x_5
0	0	1	1	0
0	1	0	1	0
0	1	0	1	1
0	1	1	0	1
1	0	1	1	1
1	1	0	1	1

.

The matrix M^1 represents the domain of the Boolean argument space where the function has the value 1. The matrix M^0 represents the domain of the Boolean argument space where the function takes the value of 0. The region in which the value of the function is undefined (the rest of the Boolean space) is denoted by the symbol M^-. The classic method of minimizing a partially defined function involves finding all the maximum intervals on the set $M^1 \cup M^-$ and covering the elements of the set M^1 with them [14].

Solution.

Minimization of the DNF of the function (M^1, M^-) and obtaining MODNF.

$F(x_1, x_2, x_3, x_4, x_5) =$

No.	x_1	x_2	x_3	x_4	x_5	f
5	0	0	1	0	1	1
7	0	0	1	1	1	1
12	0	1	1	0	0	1
18	1	0	0	1	0	1
19	1	0	0	1	1	1
22	1	0	1	1	0	1
26	1	1	0	1	0	1
28	1	1	1	0	0	1
30	1	1	1	1	0	1
0	0	0	0	0	0	–
1	0	0	0	0	1	–
2	0	0	0	1	0	–
3	0	0	0	1	1	–

No.	x_1	x_2	x_3	x_4	x_5	f
4	0	0	1	0	0	–
8	0	1	0	0	0	–
9	0	1	0	0	1	–
14	0	1	1	1	0	–
15	0	1	1	1	1	–
16	1	0	0	0	0	–
17	1	0	0	0	1	–
20	1	0	1	0	0	–
21	1	0	1	0	1	–
24	1	1	0	0	0	–
25	1	1	0	0	1	–
29	1	1	1	0	1	–
31	1	1	1	1	1	–

=

No.	x_1	x_2	x_3	x_4	x_5	f
–	0	0	1		1	1
–		1	1		0	1
–		0	0			1
–	1	0	1		0	1
–	1	1	0		0	1

=

No.	x_1	x_2	x_3	x_4	x_5	f
–	0	0			1	1
–		1	1		0	1
–		0	0			1
–	1	0			0	1
–	1		0		0	1

=

No.	x_1	x_2	x_3	x_4	x_5	f
–	0	0			1	1
–		1	1		0	1
–		0	0			1
–	1	0			0	1
–	1		0		0	1
–	1		1		0	–

=

No.	x_1	x_2	x_3	x_4	x_5	f
–	0	0			1	1
–		1	1		0	1
–		0	0			1
–	1	0			0	1
–	1				0	1

=

No.	x_1	x_2	x_3	x_4	x_5	f
–	0	0			1	1
–		1	1		0	1
–		0	0			1
–	1				0	1

=

No.	x_1	x_2	x_3	x_4	x_5	f
–	0	0			1	1
–		1	1		0	1
–		0	0			1
–	1	0			0	1
–	1				0	1

=

No.	x_1	x_2	x_3	x_4	x_5	f
–	0	0			1	1
–		1	1		0	1
–		0	0			1
–	1				0	1

. (17)

The first matrix of expression (17) represents the truth table of the partially defined DNF of the function $f(x_1, x_2, x_3, x_4, x_5)$ (M^1, M^-).

Blocks 18, 19, 0, 1, 2, 3, 16, 17, which are highlighted in red and make up the full combinatorial system of 2-(3, 8)-design, and blocks 12, 28, 30, 14, which are highlighted in blue and make up the complete combinatorial system of 2-(2, 4)-design, are subject to the operation of super-gluing of variables [13]. Blocks 5, 7 (highlighted in green) and 22, 20 (highlighted in brown), as well as blocks 26, 24 (highlighted in magenta), are subject to the operation of simple gluing of variables. The result of these logical operations is written to the second matrix of expression (17). In the second matrix of expression (17), the undefined sets of variables are not displayed because they do not participate in the further simplification of the function.

A minimal but not orthogonal DNF was obtained:

$$f_{\text{MDNF}} = \overline{x}_1\,\overline{x}_2 x_5 + x_2 x_3 \overline{x}_5 + \overline{x}_2\,\overline{x}_3 + x_1 \overline{x}_5. \qquad (18)$$

The procedure of orthogonalization of the minimal function (18) by the method of figurative transformations [16] takes the following form:

$f_{\text{MODNF}} =$

$$= \begin{array}{|cccc|}\hline x_1 & x_2 & x_3 & x_5 \\ \hline 0 & 0 & & 1 \\ & 1 & 1 & 0 \\ & 0 & 0 & \\ 1 & & & 0 \\ \hline\end{array} = \begin{array}{|cccc|}\hline x_1 & x_2 & x_3 & x_5 \\ \hline 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ & 1 & 1 & 0 \\ & 0 & 0 & \\ 1 & & & 0 \\ \hline\end{array} =$$

$$= \begin{array}{|cccc|}\hline x_1 & x_2 & x_3 & x_5 \\ \hline 0 & 0 & 1 & 1 \\ & 1 & 1 & 0 \\ & 0 & 0 & \\ 1 & & & 0 \\ \hline\end{array} = \begin{array}{|cccc|}\hline x_1 & x_2 & x_3 & x_5 \\ \hline 0 & 0 & 1 & 1 \\ & 1 & 1 & 0 \\ & 0 & 0 & 0 \\ & 0 & 0 & 1 \\ 1 & & & 0 \\ \hline\end{array} =$$

$$= \begin{array}{|cccc|}\hline x_1 & x_2 & x_3 & x_5 \\ \hline 0 & 0 & 1 & 1 \\ & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ & 0 & 0 & 1 \\ 1 & & & 0 \\ \hline\end{array} = \begin{array}{|cccc|}\hline x_1 & x_2 & x_3 & x_5 \\ \hline 0 & 0 & 1 & 1 \\ & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ & 0 & 0 & 1 \\ 1 & & & 0 \\ \hline\end{array} =$$

$$= \begin{array}{|cccc|}\hline x_1 & x_2 & x_3 & x_5 \\ \hline 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ & 0 & 0 & 1 \\ 1 & & & 0 \\ \hline\end{array} = \begin{array}{|cccc|}\hline x_1 & x_2 & x_3 & x_5 \\ \hline 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ & 0 & 0 & 1 \\ 1 & & & 0 \\ \hline\end{array}. \qquad (19)$$

Detection of redundant simple implicants in the last matrix of expression (19) is carried out using an implicant table (Table 7).

Table 7

Implicant table

No.	x_1	x_2	x_3	x_4	x_5	f	001–1	011– 0	000 – 0	– 00 – 1	1– – – 0
5	0	0	1	0	1	1	•	–	–	–	–
7	0	0	1	1	1	1	•	–	–	–	–
12	0	1	1	0	0	1	–	•	–	–	–
18	1	0	0	1	0	1	–	–	–	–	•
19	1	0	0	1	1	1	–	–	–	•	–
22	1	0	1	1	0	1	–	–	–	–	•
26	1	1	0	1	0	1	–	–	–	–	•
28	1	1	1	0	0	1	–	–	–	–	•
30	1	1	1	1	0	1	–	–	–	–	•

Behold Table 7 'til comprehending that the simple implicant «000–0» is redundant. Finally, the MODNF takes the following form:

$f_{\text{MODNF}} =$

$$= \begin{array}{|cccc|}\hline x_1 & x_2 & x_3 & x_5 \\ \hline 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ & 0 & 0 & 1 \\ 1 & & & 0 \\ \hline\end{array} = \begin{array}{|cccc|}\hline x_1 & x_2 & x_3 & x_5 \\ \hline 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ & 0 & 0 & 1 \\ 1 & & & 0 \\ \hline\end{array}. \qquad (20)$$

MODNF (20), obtained by the method of figurative transformations, contains 13 literals. The MODNF obtained on the basis of the smallest dominant independent set [14] is represented, in particular, in the form:

$f_{\text{MODNF}} =$

$$= \begin{array}{|ccccc|c|}\hline x_1 & x_2 & x_3 & x_4 & x_5 & \text{No.} \\ \hline 0 & 0 & 1 & – & 1 & 2 \\ – & 1 & 1 & – & 0 & 5 \\ 1 & 0 & – & – & 0 & 8 \\ – & 0 & 0 & – & 1 & 12 \\ \hline\end{array}. \qquad (21)$$

MODNF (21) contains 13 literals whose simple implicants are orthogonal; remaining uncovered is the defined minterm with M^1 – «11010».

Minimization of the CNF of the function (M^0, $M^{\sim}$) and obtaining MODNF.

The truth table of the partially defined CNF of the function $f(x_1, x_2, x_3, x_4, x_5)$ (M^0, $M^{\sim}$) [14] takes the following form (Table 8):

Table 8

Truth table of a partially defined CNF of the function (M^0, $M^{\sim}$)

No.	x_1	x_2	x_3	x_4	x_5	f
6	0	0	1	1	0	0
10	0	1	0	1	0	0
11	0	1	0	1	1	0
13	0	1	1	0	1	0
23	1	0	1	1	1	0
27	1	1	0	1	1	0
0	0	0	0	0	0	–
1	0	0	0	0	1	–
2	0	0	0	1	0	–
3	0	0	0	1	1	–
4	0	0	1	0	0	–
8	0	1	0	0	0	–
9	0	1	0	0	1	–
14	0	1	1	1	0	–
15	0	1	1	1	1	–
16	1	0	0	0	0	–
17	1	0	0	0	1	–
20	1	0	1	0	0	–
21	1	0	1	0	1	–
24	1	1	0	0	0	–
25	1	1	0	0	1	–
29	1	1	1	0	1	–
31	1	1	1	1	1	–

Blocks 10, 11, 13, 8, 9, 14, 15, which make up a partially balanced combinatorial system of 2-(3, 7/8)-design [13], are subject to the operation of incomplete super-gluing of variables [16]. Blocks 6, 2, and 27, 25, as well as blocks 23, 21, are subject to the operation of simple gluing of variables. The results of the specified logical operations are included in the following matrix (Fig. 5).

No.	x_1	x_2	x_3	x_4	x_5	f
–	0	0		1	0	0
–	0	1	0			0
–	0	1			1	0
–	1	0	1		1	0
–	1	1	0		1	0
–	0	1		1		–

Fig. 5. Simplification of the function (M^0, $M^{\sim}$)

In the matrix in Fig. 5 other undefined sets of variables are not shown because they do not participate in the further simplification of the function.

To continue simplifying the function, semi-gluing operations of variables are used (Fig. 6):

$$\overline{x}_1\,\overline{x}_2x_4\overline{x}_5+\overline{x}_1x_2x_4=\overline{x}_1x_4\overline{x}_5+\overline{x}_1x_2x_4;$$

$$\overline{x}_1x_2\overline{x}_3+x_1x_2\overline{x}_3x_5=\overline{x}_1x_2\overline{x}_3+x_2\overline{x}_3x_5.$$

No.	x_1	x_2	x_3	x_4	x_5	f
–	0			1	0	0
–	0	1	0			0
–	0	1			1	0
–	1	0	1		1	0
–		1	0		1	0
–	0	1		1		–

Fig. 6. Simplification of the function (M^0, $M^{\sim}$)

The final step of simplifying the function (M^0, $M^{\sim}$) is the operation of generalized gluing of variables (Fig. 7):

$$\overline{x}_1x_4\overline{x}_5+\overline{x}_1x_2x_5+\overline{x}_1x_2x_4=\overline{x}_1x_4\overline{x}_5+\overline{x}_1x_2x_5.$$

No.	x_1	x_2	x_3	x_4	x_5	f
–	0			1	0	0
–	0	1	0			0
–	0	1			1	0
–	1	0	1		1	0
–		1	0		1	0

Fig. 7. Completing the simplification of the function (M^0, $M^{\sim}$)

The detection of redundant simple implicants in the last matrix is carried out using the implicant table (Table 9).

Table 9

Implicant table

No.	x_1	x_2	x_3	x_4	x_5	f	0– –10	010– –	01– –1	101– 1	–10–1
6	0	0	1	1	0	0	•	–	–	–	–
10	0	1	0	1	0	0	•	•	–	–	–
11	0	1	0	1	1	0	–	•	•	–	–
13	0	1	1	0	1	0	–	–	•	–	–
23	1	0	1	1	1	0	–	–	–	•	–
27	1	1	0	1	1	0	–	–	–	–	•

Behold Table 9 'til comprehending that the simple implicant «010 – –» is redundant. Finally, we get MCNF:

$$f_{\text{MCNF}}=$$

x_1	x_2	x_3	x_4	x_5
0			1	0
0	1	0		
0	1			1
1	0	1		1
	1	0		1

$=$

x_1	x_2	x_3	x_4	x_5
0			1	0
0	1			1
1	0	1		1
	1	0		1

.

The transformation MCNF→MDNF→MODNF gives the minimal orthogonal disjunctive normal form. For this purpose, we open the MCNF brackets. For CNF, when transitioning from binary to algebraic form, according to Nelson's method [17], the values of the variables are inverted.

$$\left(x_1+\overline{x}_4+x_5\right)\left(x_1+\overline{x}_2+\overline{x}_5\right)\times$$

$$\times\left(\overline{x}_1+x_2+\overline{x}_3+\overline{x}_5\right)\left(\overline{x}_2+x_3+\overline{x}_5\right)=$$

$$=\left(\begin{array}{l}x_1+x_1\overline{x}_2+x_1\overline{x}_5+x_1\overline{x}_4+\\+\overline{x}_2\,\overline{x}_4+\overline{x}_4\,\overline{x}_5+x_1x_5+\overline{x}_2x_5\end{array}\right)\times$$

$$\times\left(\left(\begin{array}{l}\overline{x}_1\,\overline{x}_2+\overline{x}_1x_3+\overline{x}_1\,\overline{x}_5+x_2x_3+x_2\overline{x}_5+\\+\overline{x}_2\,\overline{x}_3+\overline{x}_3\,\overline{x}_5+\overline{x}_2\,\overline{x}_5+x_3\overline{x}_5+\overline{x}_5\end{array}\right)\right)=$$

$$=\left(x_1+\overline{x}_2\,\overline{x}_4+\overline{x}_4\,\overline{x}_5+\overline{x}_2x_5\right)\times$$

$$\times\left(\overline{x}_1\,\overline{x}_2+\overline{x}_1x_3+x_2x_3+\overline{x}_2\,\overline{x}_3+\overline{x}_5\right)=$$

$$=\left|\begin{array}{cccc}1 & & & \\ & 0 & 0 & \\ & & 0 & 0\\ & 0 & & 1\end{array}\right|\left|\begin{array}{cccc}0 & 0 & & \\ 0 & & 1 & \\ & 1 & 1 & \\ & 0 & 0 & \\ & & & 0\end{array}\right|=$$

$$=\left|\begin{array}{cccc}1 & & & \\ & & 0 & 0\\ & 0 & & 1\end{array}\right|\left|\begin{array}{ccc}0 & & 1\\ & 1 & 1\\ & 0 & 0\\ & & 0\end{array}\right|=$$

$$=\left(x_1+\overline{x}_4\,\overline{x}_5+\overline{x}_2x_5\right)\left(\overline{x}_1x_3+x_2x_3+\overline{x}_2\,\overline{x}_3+\overline{x}_5\right)=$$

$$=x_1x_2x_3+x_1\overline{x}_2\,\overline{x}_3+x_1\overline{x}_5+\overline{x}_1x_3\overline{x}_4\,\overline{x}_5+x_2x_3\overline{x}_4\,\overline{x}_5+$$

$$+\overline{x}_2\,\overline{x}_3\,\overline{x}_4\,\overline{x}_5+\overline{x}_4\,\overline{x}_5+\overline{x}_1\,\overline{x}_2x_3x_5+\overline{x}_2\,\overline{x}_3x_5=$$

$$=\left|\begin{array}{cccc}1 & 1 & 1 & \\ 1 & 0 & 0 & \\ 1 & & & 0\\ 0 & & 1 & 0\\ & 1 & 1 & 0\\ & 0 & 0 & 0\\ & & & 0\\ 0 & 0 & 1 & \\ & 0 & 0 & \end{array}\right|\left|\begin{array}{cccc}1 & 1 & 1 & \\ 1 & 0 & 0 & \\ 1 & & & 0\\ & & & 0\\ 0 & 0 & 1 & \\ & 0 & 0 & \end{array}\right|=$$

$$=\left|\begin{array}{cccc}1 & 1 & 1 & \\ 1 & 0 & 0 & \\ 1 & & & 0\\ & & 0 & 0\\ 0 & 0 & & 1\\ & 0 & 0 & 1\end{array}\right|=\left|\begin{array}{cccc}1 & 1 & 1 & \\ 1 & 0 & 0 & 0\\ 1 & 0 & 0 & 1\\ 1 & & & 0\\ & & 0 & 0\\ 0 & 0 & & 1\\ & 0 & 0 & 1\end{array}\right|=\left|\begin{array}{cccc}1 & 1 & 1 & \\ 1 & & & 0\\ & & 0 & 0\\ 0 & 0 & & 1\\ & 0 & 0 & 1\end{array}\right|. \quad (22)$$

Detection of redundant simple implicants in the last matrix of expression (22) is carried out using the implicant table (Table 10).

Table 10

Implicant table

No.	x_1	x_2	x_3	x_4	x_5	f	111– –	1– – –0	– – –00	00– –1	–00–1
5	0	0	1	0	1	1	–	–	–	•	–
7	0	0	1	1	1	1	–	–	–	•	–
12	0	1	1	0	0	1	–	–	•	–	–
18	1	0	0	1	0	1	–	•	–	–	–
19	1	0	0	1	1	1	–	–	–	–	•
22	1	0	1	1	0	1	–	•	–	–	–
26	1	1	0	1	0	1	–	•	–	–	–
28	1	1	1	0	0	1	•	•	•	–	–
30	1	1	1	1	0	1	•	•	–	–	–

From the review of Table 10 it can be seen that the simple implicant «111 – –» is redundant. Finally, we get MDNF:

$f_{\text{MDNF}} =$

x_1	x_2	x_3	x_4	x_5
1	1	1		
1				0
			0	0
0	0			1
	0	0		1

$=$

x_1	x_2	x_3	x_4	x_5
1				0
			0	0
0	0			1
	0	0		1

.

By the method of figurative transformations, we carry out the transformation of MDNF into MODNF [16]:

$f_{\text{MODNF}} =$

x_1	x_2	x_3	x_4	x_5
1				0
			0	0
0	0			1
	0	0		1

$=$

x_1	x_2	x_3	x_4	x_5
1				0
0			0	0
0	0	1		1
	0	0		1

. (23)

MODNF (23), obtained by the method of figurative transformations contains 12 literals and coincides with the MODNF obtained on the basis of the smallest maximum independent set [14].

To obtain minimal orthogonal DNFs, you can use both the method of finding the smallest dominant independent set and the method of finding the smallest maximal independent set [14]. For a partially defined Boolean function (M^1, M^0), MODNF (23) is simpler by one literal compared to MODNF (20). This means that the minimization of orthogonal DNFs should be carried out by the two specified methods and the best result should be chosen.

Possible thesauruses of methods for obtaining MODNF are given in Table 11.

Table 11

Thesauruses of methods for obtaining MODNF

No. of entry	Thesaurus for obtaining MODNF by a method that uses intervals of the argument space	Thesaurus for obtaining MODNF by the method of figurative transformations
1	Finding the smallest dominant independent set	Minimization of the DNF of a partially defined function, transformation of MDNF→MODNF
2	Finding the smallest maximum independent set	Minimization of the CNF of a partially defined function, the transformation of MCNF→MODNF

Example 5. It is required, by the method of figurative transformations, to simplify the DNF of a partially defined function $f(x_1, x_2, x_3, x_4)$ [18]:

$$f(x_1, x_2, x_3, x_4)=(1--0-10010-01--1). \quad (24)$$

Solution:

$f_{\text{MDNF}} =$

No.	x_1	x_2	x_3	x_4	f
0	0	0	0	0	1
5	0	1	0	1	1
8	1	0	0	0	1
12	1	1	0	0	1
15	1	1	1	1	1
1	0	0	0	1	–
2	0	0	1	0	–
4	0	1	0	0	–
10	1	0	1	0	–
13	1	1	0	1	–
14	1	1	1	0	–

$=$

No.	x_1	x_2	x_3	x_4	f
–			0	0	1
–		1	0	1	1
–	1	1	1		1
1	0	0	0	1	–
2	0	0	1	0	–
10	1	0	1	0	–

$=$

No.	x_1	x_2	x_3	x_4	f
–			0	0	1
–		1	0		1
–	1	1	1		1
1	0	0	0	1	–
2	0	0	1	0	–
10	1	0	1	0	–

$=$

No.	x_1	x_2	x_3	x_4	f
–			0	0	1
–		1	0		1
–	1	1			1
1	0	0	0	1	–
2	0	0	1	0	–
10	1	0	1	0	–

. (25)

The first matrix of expression (25) represents the truth table of the partially defined DNF of the function $f(x_1, x_2, x_3, x_4)$ (24).

The blocks 0, 8, 12, 4, which are highlighted in red and make up the complete combinatorial system of 2-(2, 4)-design, are subject to the operation of super-gluing of variables [13]. Blocks 5, 13 (highlighted in blue), and 15, 14 (highlighted in green) are subject to the operation of simple gluing of variables. We write the result of the specified logical operations to the second matrix of expression (25).

As a result, the minimum function is obtained:

$$f(x_1,x_2,x_3,x_4)=x_1x_2+x_2\overline{x_3}+\overline{x_3}\,\overline{x_4},$$

which contains one less inversion compared to the simplification using VBA MS Excel [18]. We note that the undefined sets of variables 1, 2, 10 were not used during the minimization of function (24). This ultimately reduced the overall complexity of simplifying the given partially defined function (24).

Example 6. It is required, by the method of figurative transformations, to simplify the partially defined Boolean function $F(x_1, x_2, x_3, x_4)$, which is given in canonical form [19]:

$$F(x_1,x_2,x_3,x_4)=$$
$$=\sum m(4,5,6,9,11,12,13,14)+\sum d(0,1,3,7). \quad (26)$$

In total, function (26) contains 8 minterms and 4 sets of undefined variables.

Solution.

The truth table of the DNF of a partially defined function $f(x_1, x_2, x_3, x_4)$ (26) takes the following form (Table 12).

Table 12

Truth table of DNF of the partially defined function $f(x_1, x_2, x_3, x_4)$ (26)

No.	x_1	x_2	x_3	x_4	f
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
9	1	0	0	1	1
11	1	0	1	1	1
12	1	1	0	0	1
13	1	1	0	1	1
14	1	1	1	0	1
0	0	0	0	0	–
1	0	0	0	1	–
3	0	0	1	1	–
7	0	1	1	1	–

Blocks 4, 6, 12, 14 and 5, 9, 13, 1, which make up the complete combinatorial system of 2-(2, 4)-design, are subject to the operation of super-gluing of variables [13]. Blocks 11, 3 are subject to the operation of simple gluing of variables. The result of these logical operations is written to the following matrix (Fig. 8).

For the final simplification of the DNF of function (26), we use the semi-gluing operation of the variables (Fig. 9).

No.	x_1	x_2	x_3	x_4	f
–			0	1	1
–		0	1	1	1
–		1		0	1
0	0	0	0	0	–
7	0	1	1	1	–

Fig. 8. Simplification of the disjunctive normal form of the function $F(x_1, x_2, x_3, x_4)$ (26)

No.	x_1	x_2	x_3	x_4	f
–			0	1	1
–		0		1	1
–		1		0	1
0	0	0	0	0	–
7	0	1	1	1	–

Fig. 9. Completing the simplification of the disjunctive normal form of the function $F(x_1, x_2, x_3, x_4)$ (26)

Further simplification of the DNF of a partially defined function (26) is no longer possible. The minimum DNF of a partially defined function (26) is:

$$f_{\text{MDNF}}=x_2\overline{x_4}+\overline{x_2}x_4+\overline{x_3}x_4. \quad (27)$$

The result of minimization (27) of the DNF of the partially defined function (26) by the method of figurative transformations (MFT) coincides with the software implementation of Quine-McCluskey in C [19], however, the MFT is significantly simpler. We note that the undefined sets of variables 0, 7 were not used during the minimization of function (26). This ultimately reduced the overall complexity of simplifying a given partially defined function.

Simplification of the CNF of the partially defined function $f(x_1, x_2, x_3, x_4)$ (Table 13).

Table 13

The truth table of the CNF of a partially defined function $f(x_1, x_2, x_3, x_4)$ (26)

No.	x_1	x_2	x_3	x_4	f
2	0	0	1	0	0
8	1	0	0	0	0
10	1	0	1	0	0
15	1	1	1	1	0
0	0	0	0	0	–
1	0	0	0	1	–
3	0	0	1	1	–
7	0	1	1	1	–

According to Nelson's method, to minimize the CNF of function (26), the values of the variables are inverted, Table 13 [17] (Fig. 10).

Blocks 2, 8, 10, 0, which make up the complete combinatorial system of 2-(2, 4)-design are subject to the operation of super-gluing of variables [13]. To blocks 15, 7, we apply the operation of simple gluing of variables. The result of these logical operations is written to the following matrix (Fig. 11).

No.	x_1	x_2	x_3	x_4	f
2	1	1	0	1	0
8	0	1	1	1	0
10	0	1	0	1	0
15	0	0	0	0	0
0	1	1	1	1	–
1	1	1	1	0	–
3	1	1	0	0	–
7	1	0	0	0	–

Fig. 10. Simplification of the conjunctive normal form of the function $F(x_1, x_2, x_3, x_4)$ (26)

No.	x_1	x_2	x_3	x_4	f
–		1		1	0
–		0	0	0	0
1	1	1	1	0	–
3	1	1	0	0	–

Fig. 11. Completing the simplification of the conjunctive normal form of the function $F(x_1, x_2, x_3, x_4)$ (26)

Further simplification of the CNF of the partially defined function (26) is no longer possible. The minimal CNF of the partially defined Boolean function (26) takes the form:

$$f_{\text{MCNF}} = (x_2 + x_4)(\overline{x_2} + \overline{x_3} + \overline{x_4}). \quad (28)$$

Expression (28) contains one literal less than expression (27), and therefore the MCNF (28) is simpler compared to MDNF (27). We note that the undefined sets of variables 1, 3 were not used during the minimization of the CNF of function (26). This ultimately reduced the overall complexity of simplifying the CNF of a given function.

Example 7. It is required, by the method of figurative transformations, to simplify the DNF of a partially defined logical function $f(x_1, x_2, x_3, x_4)$, which is given by the Weich diagram [20]:

X1 \ X2	-	-	0	1
-	-	1	-	0
0	0	0	0	-
-	-	1	-	1

X3 (rows); X4 (columns)

Fig. 12. Partially defined Boolean function $f(x_1, x_2, x_3, x_4)$ (Weich diagram)

Solution.

The Weich diagram for four variables takes the following form (Fig. 13).

X1 \ X2	1100	1101	1001	1000
1110	1111	1011	1010	
0110	0111	0011	0010	
0100	0101	0001	0000	

X3 (rows); X4 (columns)

Fig. 13. Weich diagram for four variables

$$f_{\text{DeDNF}}(x_1, x_2, x_3, x_4) =$$

No.	x_1	x_2	x_3	x_4	f
0	0	0	0	0	1
5	0	1	0	1	1
8	1	0	0	0	1
15	1	1	1	1	1
1	0	0	0	1	–
2	0	0	1	0	–
4	0	1	0	0	–
11	1	0	1	1	–
12	1	1	0	0	–
13	1	1	0	1	–
14	1	1	1	0	–

$=$

No.	x_1	x_2	x_3	x_4	f
0	0		0		1
8	1	0	0	0	1
15	1	1			1
2	0	0	1	0	–
11	1	0	1	1	–

$=$

No.	x_1	x_2	x_3	x_4	f
0	0		0		1
8	1		0	0	1
15	1	1			1
2	0	0	1	0	–
11	1	0	1	1	–

$=$

No.	x_1	x_2	x_3	x_4	f
0	0		0		1
8	1		0	0	1
15	1	1			1
2	0	0	1	0	–
11	1	0	1	1	–

$=$

No.	x_1	x_2	x_3	x_4	f
0	0		0		1
8			0	0	1
15	1	1			1
2	0	0	1	0	–
11	1	0	1	1	–

. (29)

The first matrix of expression (29) represents the truth table of the partially defined DNF of the function $f(x_1, x_2, x_3, x_4)$ (Fig. 12). Undefined sets of variables «0010», «1011» of the last matrix of expression (29) do not participate in further minimization. Then the algebraic representation for the last matrix of expression (29) will take the form:

$$f_{\text{DeDNF}}(x_1, x_2, x_3, x_4) = x_1 x_2 + \overline{x_1}\,\overline{x_3} + \overline{x_3}\,\overline{x_4}, \quad (30)$$

which coincides with [20].

In an attempt to optimize the dead-end DNF (30), we shall carry out all possible operations of generalized gluing of variables, followed by the use of an implicant table (Table 14) to identify simple implicants with fewer inversions:

$$f_{\text{DeDNF}}(x_1, x_2, x_3, x_4) =$$

x_1	x_2	x_3	x_4
0		0	
		0	0
1	1		

$=$

x_1	x_2	x_3	x_4
	1	0	
0		0	
		0	0
1	1		

.

From the review of Table 14 it can be seen that among the simple implicants – 10 –, 0 – 0 –, – – 00, and 11 – –, it is necessary to choose – 10 –, – – 00 and 11 – – because they contain fewer inversions.

Table 14

Implicant table of DNF of the function f(x1, x2, x3, x4) (Fig. 12)

No.	x_1	x_2	x_3	x_4	f	–10 –	0 – 0 –	– – 00	11– –
0	0	0	0	0	1	–	•	•	–
5	0	1	0	1	1	•	•	–	–
8	1	0	0	0	1	–	–	•	–
15	1	1	1	1	1	–	–	–	•

Then the MDNF of the function (Fig. 12) will take the form:

$$f_{\text{MDNF}}(x_1,x_2,x_3,x_4)=x_1x_2+x_2\overline{x_3}+\overline{x_3}\,\overline{x_4}. \quad (31)$$

The minimal DNF (31) contains one less inverter compared to (30) and passes verification for the DNF and CNF of the given function (Fig. 12).

Example 8. It is required to simplify the DNF of a partially defined function $f(x_1, x_2, x_3, x_4)$, which is given as follows [21]:

$$f(x_1,x_2,x_3,x_4)= \underset{1}{=\vee}\left(0,1^*,2^*,4^*,6,7^*,8^*,9,11^*,13^*,14,15^*\right). \quad (32)$$

In expression (32), the decimal numbers of the sets on which the function is not defined are represented with a sign ≪*≫.

Solution:

$f_{\text{DeDNF}}(x_1,x_2,x_3,x_4)=$

No.	x_1	x_2	x_3	x_4	f
0	0	0	0	0	1
6	0	1	1	0	1
9	1	0	0	1	1
14	1	1	1	0	1
1	0	0	0	1	–
2	0	0	1	0	–
4	0	1	0	0	–
7	0	1	1	1	–
8	1	0	0	0	–
11	1	0	1	1	–
13	1	1	0	1	–
15	1	1	1	1	–

$$=\overline{x_1}\,\overline{x_2}\,\overline{x_3}\,\overline{x_4}+\overline{x_1}x_2x_3\overline{x_4}+x_1\overline{x_2}\,\overline{x_3}x_4+x_1x_2x_3\overline{x_4}+$$
$$+\overline{x_1}\,\overline{x_2}\,\overline{x_3}x_4+\overline{x_1}x_2x_3x_4+x_1\overline{x_2}\,\overline{x_3}\,\overline{x_4}+x_1x_2x_3x_4=$$
$$=\overline{x_2}\,\overline{x_3}\left(\overline{x_1}\,\overline{x_4}+x_1x_4\right)+x_2x_3\left(\overline{x_1}\,\overline{x_4}+x_1x_4\right)+$$
$$+x_2x_3\left(x_1\overline{x_4}+\overline{x_1}x_4\right)+\overline{x_2}\,\overline{x_3}\left(x_1\overline{x_4}+\overline{x_1}x_4\right)=$$
$$=\left(\overline{x_1}\,\overline{x_4}+x_1x_4\right)\left(\overline{x_2}\,\overline{x_3}+x_2x_3\right)+$$
$$+\left(x_1\overline{x_4}+\overline{x_1}x_4\right)\left(\overline{x_2}\,\overline{x_3}+x_2x_3\right)=$$
$$=\left(\overline{x_2}\,\overline{x_3}+x_2x_3\right)\left(\left(\overline{x_1}\,\overline{x_4}+x_1x_4\right)+\left(x_1\overline{x_4}+\overline{x_1}x_4\right)\right)=$$
$$=\left(\overline{x_2}\,\overline{x_3}+x_2x_3\right)\left(\overline{x_1}\,\overline{x_4}+x_1x_4+x_1\overline{x_4}+\overline{x_1}x_4\right)=$$
$$=\left(\overline{x_2}\,\overline{x_3}+x_2x_3\right)\left(\overline{x_1}\left(\overline{x_4}+x_4\right)+x_1\left(x_4+\overline{x_4}\right)\right)=$$
$$=\overline{x_2}\,\overline{x_3}+x_2x_3=\overline{x_2\oplus x_3}. \quad (33)$$

The matrix of expression (33) represents the truth table of a partially defined DNF of the function $f(x_1, x_2, x_3, x_4)$ (32). In turn, the DNF of the partially defined function (32) is singular [16]. In this regard, the minimum function can be obtained in a polynomial basis:

$$f_{\text{MPNF}}=\overline{x_2\oplus x_3}. \quad (34)$$

The expression $\overline{x}_2\overline{x}_3+x_2x_3$ represents the minimum function in the main basis, coinciding with [21].

Thus, the minimum function (34) in the polynomial basis is two literals simpler than the minimum function of the main basis.

5. 5. Minimization of partially defined Boolean functions in the Reed-Muller basis

Example 9. It is required to minimize the partially defined Boolean function $f(x_1, x_2, x_3, x_4)$ in the Reed-Muller basis [22] using the method of figurative transformations:

$$\begin{cases} Y^1=\{3,5,6,9,12,15\}^1, \\ Y^{\sim}=\{1,2,8,11\}^{\sim}, \end{cases} \quad (35)$$

where the symbols Y^1 and $Y^{\sim}$ denote sets of binary minterms and sets of undefined variables of a partially defined function $f(x_1, x_2, x_3, x_4)$, respectively [22].

Solution.

In the ideal polynomial set-theoretic form (PSTF) [22], function (35) will take the following form:

$$\begin{cases} Y^{\oplus}=\{(0011),(0101),(0110),(1001),(1100),(1111)\}^{\oplus}; \\ Y^{\tilde{\oplus}}=\{(0001),(0010),(1000),(1011)\}^{\tilde{\oplus}}. \end{cases} \quad (36)$$

Since function (36) is singular [16], minimization (36) can be performed in DNF or PNF. The simplest procedure for minimizing function (36) in DNF with the transition to a mixed basis [16] and with the subsequent transition to the Reed-Muller basis. Minimization of the DNF of a partially defined function (36) (Table 15).

Blocks 3, 9, 1, 11, which make up the complete combinatorial system of 2-(2, 4)-design are subject to the operation of super-gluing of variables [13]. To blocks 6, 2 and 12, 8, the operation of simple gluing of variables is applied. The result of these logical operations is written to the following matrix (Fig. 14).

Table 15

Truth table of the DNF of a partially defined function $f(x_1, x_2, x_3, x_4)$ (36)

No.	x_1	x_2	x_3	x_4	f
3	0	0	1	1	1
5	0	1	0	1	1
6	0	1	1	0	1
9	1	0	0	1	1
12	1	1	0	0	1
15	1	1	1	1	1
1	0	0	0	1	–
2	0	0	1	0	–
8	1	0	0	0	–
11	1	0	1	1	–

No.	x_1	x_2	x_3	x_4	f
–		0		1	1
–	0	1	0	1	1
–	0		1	0	1
–	1		0	0	1
–	1	1	1	1	1

Fig. 14. Simplification of the disjunctive normal form of the function $f(x_1, x_2, x_3, x_4)$ (36)

The final step in simplifying function (36) is the operation of semi-gluing of variables (Fig. 15).

No.	x_1	x_2	x_3	x_4	f
–		0		1	1
–	0		0	1	1
–	0		1	0	1
–	1		0	0	1
–	1		1	1	1

Fig. 15. Completion of simplification of the disjunctive normal form of function $f(x_1, x_2, x_3, x_4)$ (36)

Further simplification of the DNF of function (36) is no longer possible. The abbreviated function (36) in a mixed basis is:

$$f_{\text{abbreviated}} = \overline{x_1}(x_3 \oplus x_4) + x_1\overline{(x_3 \oplus x_4)} + \overline{x_2}x_4 =$$
$$= x_1 \oplus x_3 \oplus x_4 + \overline{x_2}x_4.$$

To obtain the minimum function in the Reed-Muller basis, we apply identity (6):

$$f_{\text{MPNF}} =$$
$$= \overline{x_1}(x_3 \oplus x_4) + x_1\overline{(x_3 \oplus x_4)} + \overline{x_2}x_4 =$$
$$= x_1 \oplus x_3 \oplus x_4 + \overline{x_2}x_4 =$$
$$= x_1 \oplus x_3 \oplus x_4 \oplus \overline{x_2}x_4 = x_1 \oplus x_3 \oplus x_2x_4. \quad (37)$$

The result of minimizing (37) the DNF of a partially defined function (36) by the method of figurative transformations coincides with [22].

Minimization of PNF of a partially defined function (36):

$$f_{\text{MPNF}}(x_1, x_2, x_3, x_4) =$$

No.	x_1	x_2	x_3	x_4	f
3	0	0	1	1	1
5	0	1	0	1	1
6	0	1	1	0	1
9	1	0	0	1	1
12	1	1	0	0	1
15	1	1	1	1	1
1	0	0	0	1	–
2	0	0	1	0	–
8	1	0	0	0	–
11	1	0	1	1	–

=

No.	x_1	x_2	x_3	x_4	f
3		0		1	1
5	0	1	0	1	1
6	0		1	0	1
12	1		0	0	1
15	1	1	1	1	1

=

No.	x_1	x_2	x_3	x_4	f
3		0		1	1
5		1	0	1	1
6			1	0	1
12	1			0	1
15	1	1		1	1

=

No.	x_1	x_2	x_3	x_4	f
3				1	1
5		1	1	1	1
6			1	0	1
12	1				1
15	1	0		1	1

=

No.	x_1	x_2	x_3	x_4	f
3				1	1
5		0	1	1	1
6			1		1
12	1				1
15	1	0		1	1

. (38)

The first matrix of expression (38) represents the truth table of the partially defined DNF of the function $f(x_1, x_2, x_3, x_4)$ (36).

Blocks 3, 9, 1, 11, which are highlighted in red and make up the full combinatorial system of 2-(2, 4)-design, are subject to the operation of super-gluing of variables [13]. To blocks 6, 2 (highlighted in blue) and blocks 12, 8 (highlighted in green), the operation of simple gluing of variables is applied. The result of these logical operations is reflected in the second matrix of expression (38). As a result, a shortened function in the Reed-Muller basis was obtained:

$$f_{abbreviated} = x_1 \oplus x_3 \oplus x_4 \oplus x_1\overline{x_2}x_4 \oplus \overline{x_2}x_3x_4. \quad (39)$$

To minimize the reduced function (39), we apply identity (7):

$$f_{\text{MPNF}} =$$
$$= x_1 \oplus x_3 \oplus x_4 \oplus x_1\overline{x_2}x_4 \oplus \overline{x_2}x_3x_4 =$$
$$= x_1 \oplus x_3 \oplus x_4 + \overline{x_2}x_4(x_1 \oplus x_3) =$$
$$= x_1 \oplus x_3 \oplus x_4 \oplus \overline{x_2}x_4 = x_1 \oplus x_3 \oplus x_2x_4. \quad (40)$$

The result of minimizing (40) PNF of a partially defined function (36) by the method of figurative transformations coincides with [22].

6. Discussion of results of minimization of partially defined Boolean functions by the method of figurative transformations

The mathematical apparatus for minimizing Boolean functions by the method of figurative transformations was considered in works [10, 12, 13, 16, 17, 23, 24], etc.

The technology of minimizing Boolean functions using figurative transformations is given in Table 16.

Table 16

Technology for minimizing Boolean functions using figurative transformations

1	Binary combinatorial systems with repeated 2-(n, b)-design, 2-(n, x/b)-design
2	Verbal and figurative representation of information
3	The logical operation of super-gluing variables
4	Logical operation of incomplete super-gluing of variables
5	Hermeneutics of logical operations on binary equivalents of logical functions
6	Protocols of figurative transformations
7	A sign of the minimum logical function,
8	Minimizing Boolean functions on a complete truth table
9	Algorithm of the analytical method and its automation
10	The spread of the analytical method to other logical bases
11	Algebra of equivalent transformations in the class of perfect normal forms of functions of Schaeffer algebra
12	Algebra of equivalent transformations in the class of perfect implicative normal forms
13	Algorithms for simplifying Boolean functions using logical operations of absorption and super-gluing of variables
14	Logic operations stack
15	Algorithms for simplifying the PNF of a Boolean function by inserting the same conjuncterms, followed by a super-gluing operation of variables
16	Singular function
17	Algebra of equivalent transformations in the class of polynomial normal forms of Boolean functions
18	Mixed basis

New components of minimization for partially defined Boolean functions are given in Table 17.

Table 17

New components of minimization technology using figurative transformations for partially defined Boolean functions

1	Combining a sequence of logical operations of super- and simple gluing of variables with the possible use of an implicant table to detect unnecessary simple implicants
2	Dead-end DNF can be simplified by carrying out all operations of generalized gluing of variables, followed by the use of an implicant table: – to identify unnecessary simple implicants, – to select simple implications with a minimum number of inversions
3	Identification of the location of equivalent transformations using combinatorial systems of 2-(n, b)-design, 2-$(n, x/b)$-design

Table 18 gives the results of minimizing partially defined Boolean functions borrowed from the works of other authors and by the method of figurative transformations.

Table 18

Comparative table of examples of minimization of partially defined Boolean functions borrowed from the works of other authors and by the method of figurative transformations

Example number	Number of input variables	The name of the minimization method	The result of minimization	Method of figurative transformations
3	6	Heuristic method [15]	14 literals	12 literals
4	5	Method of finding the smallest dominant independent set [14]	MODNF 13 literals (not all defined minterms are covered)	MODNF 13 literals
4	5	Method for finding the smallest maximum independent set [14]	The results of minimization are the same	
5	4	VBA MS Excel [18]	4 inversions	3 inversions
6	4	Software implementation of Quine-McCluskey in C [19]	MDNF 6 literals	MDNF 6 literals MCNF 5 literals
7	4	Weich diagram [20]	4 inversions	3 inversions
8	4	Quine-McCluskey method [21]	MDNF 4 literals	MDNF 4 literals MCNF 2 literals
9	4	Conjunctterm decoupling method [22]	The results of minimization are the same	

Table 18 gives a representative sample of examples of simplification of partially defined Boolean functions. The method of figurative transformations demonstrates the best or the same result.

The construction of a strictly minimal DNF for an arbitrary Boolean function from n variables is a challenging combinatorial problem, which is practically solved only for a relatively small n (maximum 12) [15]. There are some methods, such as the Carnot map, which become difficult to consider when the number of variables is taken more than six, and the Quaint-McCluskey method, which overcomes the shortcomings of Carnot maps but becomes complex with a large number of variables. The search time for the optimal function, in this case, increases by 2^{2n}, where n is the bit depth of the Boolean function. In this regard, approximate methods, in particular heuristic, have become widespread, which makes it possible to use a computer to find close to the optimal solution in an acceptable time [25]. However, traditional heuristic-based logical synthesis has many problems as computing power continues to grow and new computational paradigms emerge. With an increase in computing power, logical optimization is increasingly looking for accurate solutions, rather than suboptimal ones [26].

A new graphical method suitable for minimizing the logical functions of five or more variables is proposed in [27]. Paper [28] reports a set of rules that simplify minimization using the Xiao map [27] and demonstrates that the Xiao map method has advantages over the Quine–McCluskey algorithmic method.

Example 10. It is required, by the method of figurative transformations, to simplify the 6-bit Boolean function F(A,B,C,D,E,F), which is given in canonical form [28]:

$$F(\mathrm{A,B,C,D,E,F}) = \Sigma m \begin{pmatrix} 0,2,5,6,8,10,14,16,17,18, \\ 21,22,24,26,30,34,37,38, \\ 42,46,49,50,53,54,58,62 \end{pmatrix},$$

Solution:

$F(\mathrm{A,B,C,D,E,F}) =$

$=$

No.	A	B	C	D	E	F
0	0	0	0	0	0	0
2	0	0	0	0	1	0
5	0	0	0	1	0	1
6	0	0	0	1	1	0
8	0	0	1	0	0	0
10	0	0	1	0	1	0
14	0	0	1	1	1	0
16	0	1	0	0	0	0
17	0	1	0	0	0	1
18	0	1	0	0	1	0
21	0	1	0	1	0	1
22	0	1	0	1	1	0
24	0	1	1	0	0	0

No.	A	B	C	D	E	F
26	0	1	1	0	1	0
30	0	1	1	1	1	0
34	1	0	0	0	1	0
37	1	0	0	1	0	1
38	1	0	0	1	1	0
42	1	0	1	0	1	0
46	1	0	1	1	1	0
49	1	1	0	0	0	1
50	1	1	0	0	1	0
53	1	1	0	1	0	1
54	1	1	0	1	1	0
58	1	1	1	0	1	0
62	1	1	1	1	1	0

$=$

No.	A	B	C	D	E	F
0	0			0	0	0
2					1	0
5			0	1	0	1
17		1	0	0	0	1

$=$

No.	A	B	C	D	E	F
0	0			0		0
2					1	0
5			0	1	0	1
17		1	0		0	1

. (41)

The result of simplifying (41) the 6-bit Boolean function F(A,B,C,D,E,F) coincides with [28] but the simplification technique of figurative transformations is simpler.

A feature of the proposed solutions is the use of combinatorial systems of 2-(n, b)-design, 2-$(n, x/b)$-design within the truth table of a given partially defined Boolean func-

tion (PDBF). In contrast to the representation of PDBF as a set of fully defined Boolean functions, which are obtained using the appropriate redefinition (sorting through all possible substitutions 0 or 1 instead of «–»), the use of the specified combinatorial systems makes it possible to reduce the number of ways to identify PDBF. Such a reduction is possible due to the interpretation by the systems of 2-(n, b)-design, 2-(n, x/b)-design of logical operations for the equivalent transformation of PDBF. In this regard, it becomes possible to determine such a binary configuration within the truth table of a given PDBF, which provides an implication algorithm for simplifying the function, through logical operations of super-gluing of variables and/or simple gluing of variables. The established configuration contains the defined sets of variables and some of the indefinite sets of variables, in particular. With this technique of simplifying PDBF, not necessarily all indefinite sets of variables will be required. Thus, the principle of minimization of PDBF is established, which reduces the complexity and increases the productivity of the procedure for minimizing a given PDBF, compared in particular with the transformation of the problem from the field of Boolean algebra into the classical algebraic domain, methods for finding approximate Boolean schemes, large training sets of examples with a large number of primary input data, ways to restore the accuracy of the logical scheme, machine learning techniques.

The application of the obtained result makes it possible to improve and expand the technology of designing electronic components and devices for their use in digital technologies, which are based on the use of Boole and Reed-Muller logical bases.

The visual form of 2-dimensional binary matrices allows for a manual way to simplify partially defined Boolean functions using a mathematical editor, for example, Math Type 7.4.0 (USA): examples 1, 2, 3, 4 (minimization of DNF), 5, 7, 8, 9 (minimization of PNF), or using MS Word tables: examples 4 (minimization of CNF), 6, 9 (minimization of DNF).

The use of MFT to minimize partially defined functions in the Boolean and Reed-Muller bases deduces, to a certain extent, the problem of simplifying partially defined Boolean functions to the level of a well-researched problem in the class of disjunctive-conjunctive normal forms (DCNF) of Boolean functions. The limitation of using the method of figurative transformations are cases when the switching function is represented in a mixed basis. In this case, the function must be represented by one logical basis.

The weak side of the method under consideration is in its small practical application to minimize partially defined Boolean functions, followed by the design and manufacture of appropriate computational components. The negative internal factors of MFT are associated with additional time spent on establishing protocols for simplifying partially defined logical functions in the Boolean and Reed-Muller bases, followed by the creation of a library of protocols illustrating the corresponding figurative transformations.

The prospect of further research may be the development of a synergistic method for simplifying Boolean functions based on the visual-matrix form of representation.

7. Conclusions

1. With the optimal combination of the sequence of logical operations of super-gluing and simple gluing of variables in the initial truth table, redundant simple implicants can be detected using an implicant table. The effectiveness of this procedure is demonstrated by the following examples: example 3 – minimization of a 6-bit Boolean function, example 4 – minimization of a 5-bit Boolean function, example 8 – minimization of a 4-bit Boolean function. Minimization efficiency in these examples gives grounds for choosing in favor of using a sequence of logical operations of super-gluing and simple gluing of variables. Thus, the 2-(n, b)-design system and the sequential alternation of logical operations of super-gluing of variables (if such an operation is possible) and simple gluing of variables in the first, and, in some cases, in the second binary matrix, with the possible use of an implicant table to detect unnecessary simple implicants, provides unambiguousness and sufficient efficiency of the algorithm for minimizing Boolean functions, including partially defined Boolean functions.

2. Combinatorial systems of 2-(n, b)-design, 2-(n, x/b)-design, which are part of the binary structures of truth tables, provide unambiguous detection of the location of equivalent transformations to simplify Boolean functions. The set place of equivalent transformations implicates the algorithm for minimizing Boolean functions. Thus, the 2-(n, b)-design, 2-(n, x/b)-design systems have an information capacity, which makes it possible to replace the binary redefinition of partially defined Boolean functions with an effect, such as in Fig. 2, with the method of figurative transformations. This, in turn, makes it possible to reduce complexity and speed up the minimization procedure without loss of functionality, compared to brute force binary redefinitions of partially defined Boolean functions. The interpretation of the result is that the 2-(n, b)-design, 2-(n, x/b)-design systems are a reflection of logical operations. Therefore, the detection of combinatorial systems in the truth table of Boolean functions directly and unambiguously indicates logical operations for equivalent transformations of Boolean expressions.

3. The differences in the result of minimizing a partially defined Boolean function by six variables by the heuristic method and the method of figurative transformations are demonstrated by example 3 – minimization of a 6-bit Boolean function. The cost of implementing the minimum function obtained by the method of figurative transformations is: 4 conjuncterms, 12 literals, 6 inverters. The cost of implementing the minimum function obtained by the heuristic method is: 5 conjuncterms, 14 literals, 8 inverters. Thus, the implementation of the minimum function obtained by the method of figurative transformations is simpler.

4. The effectiveness of the method of figurative transformations to minimize partially defined Boolean functions in the main basis is demonstrated by the following examples:

– Example 4 – minimizing a 5-bit partially defined Boolean function. The results of minimizing the CNF of a given function are the same. In the compared example, two methods are used to obtain a minimum orthogonal DNF. One of them reduces this task to obtaining the smallest dominant set in a graph by covering its vertices with their closed circles, the other to obtaining the maximum independent set via a lexicographic brute force. The method of figurative transformations uses a table of truth of a given function and the transformation of the obtained minimum function into a minimal orthogonal DNF. The mathematical apparatus and the technique of minimizing the method of figurative transformations is simpler.

– Example 5 – minimizing a 4-bit partially defined Boolean function. The cost of implementing the minimum function

obtained by the method of figurative transformations is one less inverter. In the compared example, the software in the form of an MS Excel spreadsheet is used. The method of figurative transformations, to a certain extent, does not use automation to simplify functions, and is therefore simpler.

– Example 6 – minimizing a 4-bit partially defined Boolean function. The results of minimizing the DNF of a given function are the same. The compared example uses software written in C. Thus, minimizing a given function by the method of figurative transformations is much simpler.

– Example 7 – minimizing a 4-bit partially defined Boolean function. The cost of implementing the minimum function obtained by the method of figurative transformations is one less inverter. The result is achieved by carrying out all logical operations of generalized gluing of variables in the dead-end disjunctive normal form of a given function.

– Example 8 – minimizing a 4-bit partially defined Boolean function. The results of minimizing the DNF of a given function are the same for the main basis. The cost of implementing the minimum function in the polynomial basis obtained by the method of figurative transformations is two literals less. Thus, the minimum function in a polynomial basis is two literals simpler than the minimum function of the main basis.

5. The polynomial basis algebraic apparatus $\{\wedge, \oplus, 1\}$ makes it possible to introduce a method of figurative transformations to minimize partially defined Boolean functions. Since a partially defined Boolean function can be singular, this makes it possible to choose the basis of minimization – basic or polynomial. The ability to choose a logical basis expands the options for simplifying the Boolean function, which increases the effectiveness of the procedure for minimizing partially defined Boolean functions.

The result of minimizing a partially defined Boolean function by four variables by the method of decoupling conjuncterms and by the method of figurative transformations is demonstrated by example 9 – minimization of a 4-bit partially defined Boolean function. The results of minimizing the PNF of a given function are the same. Minimization of partially defined Boolean functions by the method of decoupling conjuncterms consists of two stages. At the 1st stage, the procedure for breaking the specified conjuncterms using a splitting matrix is implemented and a set of coverage is obtained. At the 2nd stage, the procedure of iterative simplification of pairs of conjuncterms of the coverage set obtained at stage 1 is carried out on the basis of generalized rules of theorems and a minimum function is formed for a given partially defined function f. The method of figurative transformations to minimize the PNF of partially defined Boolean functions uses a truth table of a given function and the rules of equivalent transformations in a polynomial basis.

Conflicts of interest

The authors declare that they have no conflicts of interest in relation to the current study, including financial, personal, authorship, or any other, that could affect the study and the results reported in this paper.

Funding

The study was conducted without financial support.

Data availability

All data are available in the main text of the manuscript.

References

1. Savel'ev, A. Ya. (1987). Prikladnaya teoriya cifrovyh avtomatov. Moscow: Vysshaya shkola, 272. Available at: https://vdoc.pub/documents/-4o35jbu52gg0
2. Prihozhiy, A. A. (2013). Chastichno opredelyonnye logicheskie sistemy i algoritmy. Minsk: BNTU, 343. Available at: https://rep.bntu.by/handle/data/37237
3. Papakonstantinou, K. G., Papakonstantinou, G. (2018). A Nonlinear Integer Programming Approach for the Minimization of Boolean Expressions. Journal of Circuits, Systems and Computers, 27 (10), 1850163. doi: https://doi.org/10.1142/s0218126618501633
4. Fišer, P., Hlavička, J. (2000). Efficient minimization method for Incompletely defined Boolean functions. Conference: 4th Int. Workshop on Boolean Problems (IWSBP). Available at: https://www.researchgate.net/publication/260987269_Efficient_minimization_method_for_incompletely_defined_Boolean_functions
5. Dimopoulos, A. C., Pavlatos, C., Papakonstantinou, G. (2022). Multi-output, multi-level, multi-gate design using non-linear programming. International Journal of Circuit Theory and Applications, 50 (8), 2960–2968. doi: https://doi.org/10.1002/cta.3300
6. Scholl, C., Melchior, S., Hotz, G., Molitor, P. (1997). Minimizing ROBDD sizes of incompletely specified Boolean functions by exploiting strong symmetries. Proceedings European Design and Test Conference. ED & TC 97. doi: https://doi.org/10.1109/edtc.1997.582364
7. Rytsar, B. (2015). The Minimization Method of Boolean Functions in Polynomial Set-theoretical Format. Conference: Proc. 24th Inter. Workshop, CS@P'2015. Rzeszow, 130–146. Available at: https://www.researchgate.net/publication/298158364_The_Minimization_Method_of_Boolean_Functionns_in_Polynomial_Set-theoretical_Format
8. Costamagna, A., De Micheli, G. (2023). Accuracy recovery: A decomposition procedure for the synthesis of partially-specified Boolean functions. Integration, 89, 248–260. doi: https://doi.org/10.1016/j.vlsi.2022.12.008
9. Boroumand, S., Bouganis, C.-S., Constantinides, G. A. (2021). Learning Boolean Circuits from Examples for Approximate Logic Synthesis. Proceedings of the 26th Asia and South Pacific Design Automation Conference. doi: https://doi.org/10.1145/3394885.3431559
10. Solomko, M. (2021). Developing an algorithm to minimize boolean functions for the visual-matrix form of the analytical method. Eastern-European Journal of Enterprise Technologies, 1 (4 (109)), 6–21. doi: https://doi.org/10.15587/1729-4061.2021.225325

11. Riznyk, V., Solomko, M., Tadeyev, P., Nazaruk, V., Zubyk, L., Voloshyn, V. (2020). The algorithm for minimizing Boolean functions using a method of the optimal combination of the sequence of figurative transformations. Eastern-European Journal of Enterprise Technologies, 3 (4 (105)), 43–60. doi: https://doi.org/10.15587/1729-4061.2020.206308
12. Minimizatsiya nepovnistiu vyznachenykh lohichnykh funktsiy. Available at: https://studfile.net/preview/14499737/page:17/
13. Riznyk, V., Solomko, M. (2017). Application of super-sticking algebraic operation of variables for Boolean functions minimization by combinatorial method. Technology Audit and Production Reserves, 6 (2 (38)), 60–76. doi: https://doi.org/10.15587/2312-8372.2017.118336
14. Pottosin, Yu. V. (2021). Minimization of Boolean functions in the class of orthogonal disjunctive normal forms. Informatics, 18 (2), 33–47. doi: https://doi.org/10.37661/1816-0301-2021-18-2-33-47
15. Zakrevskij, A. D., Toropov, N. R., Romanov, V. I. (2010). DNF-implementation of partial boolean functions of many variables. Informatics, 1 (25), 102–111. Available at: https://inf.grid.by/jour/article/view/461/419
16. Solomko, M., Batyshkina, I., Khomiuk, N., Ivashchuk, Y., Shevtsova, N. (2021). Developing the minimization of a polynomial normal form of boolean functions by the method of figurative transformations. Eastern-European Journal of Enterprise Technologies, 2 (4 (110)), 22–37. doi: https://doi.org/10.15587/1729-4061.2021.229786
17. Riznyk, V., Solomko, M. (2018). Minimization of conjunctive normal forms of boolean functions by combinatorial method. Technology Audit and Production Reserves, 5 (2 (43)), 42–55. doi: https://doi.org/10.15587/2312-8372.2018.146312
18. Sdvizhkov, O. A. (2012). Diskretnaya matematika i matematicheskie metody ekonomiki s primeneniem VBA Ehcel. Moscow: DMK, 212. Available at: https://www.studmed.ru/sdvizhkov-o-a-diskretnaya-matematika-i-matematicheskie-metody-ekonomiki-s-primeneniem-vba-excel_9edfd48c895.html
19. Huang, J. (2014). Programing implementation of the Quine-McCluskey method for minimization of Boolean expression. arXiv. doi: https://doi.org/10.48550/arXiv.1410.1059
20. Matematychna lohika ta dyskretna matematyka (2020). Kremenchuk, 61. Available at: http://document.kdu.edu.ua/metod/2020_2182.pdf
21. Novytskyi, I. V., Us, S. A. (2013). Dyskretna matematyka v prykladakh i zadachakh. Dnipropetrovsk, 89. Available at: https://sau.nmu.org.ua/ua/osvita/metod/Discrete_Math(Novitskiy_Us_NMU_SAU).pdf
22. Rytsar, B. Ye. (2015). A New Method of Minimization of Logical Functions in the Polynomial Set-theoretical Format. 2. Minimization of Complete and Incomplete Functions. УСиМ, 4, 9–30. Available at: http://dspace.nbuv.gov.ua/handle/123456789/87235
23. Solomko, M., Batyshkina, I., Voitovych, I., Zubyk, L., Babych, S., Muzychuk, K. (2020). Devising a method of figurative transformations for minimizing boolean functions in the implicative basis. Eastern-European Journal of Enterprise Technologies, 6 (4 (108)), 32–47. doi: https://doi.org/10.15587/1729-4061.2020.220094
24. Solomko, M., Tadeyev, P., Zubyk, L., Babych, S., Mala, Y., Voitovych, O. (2021). Implementation of the method of figurative transformations to minimizing symmetric Boolean functions. Eastern-European Journal of Enterprise Technologies, 4 (4 (112)), 23–39. doi: https://doi.org/10.15587/1729-4061.2021.239149
25. Zakrevskiy, A. D. (1981). Logicheskiy sintez kaskadnyh shem. Moscow, 414.
26. Chu, Z., Pan, H. (2023). Survey on Exact Logic Synthesis Based on Boolean SATisfiability. Journal of Electronics & Information Technology, 45 (1), 14–23. doi: https://doi.org/10.11999/JEIT220391
27. Yong-Xin, X. (1987). Xiao map for minimization of boolean expression. International Journal of Electronics, 63 (3), 353–358. doi: https://doi.org/10.1080/00207218708939138
28. Osuagwu, C. C., Anyanwu, C. D., Agada, J. O. (1989). Fast Minimization on the Xiao Map Using Row Group Structure Rules. Nigerian Journal of Technology, 13 (1), 51–61. Available at: https://www.ajol.info/index.php/njt/article/view/123260