# Information modeling of the normal forms of logical functions in Maple system

*Aleksandr* Olenev[1*], *Ksenia* Kirichek[1], *Aleksandr* Shuvaev[2], and *Elena* Petlina[1]

[1]Stavropol State Pedagogical Institute, 417A Lenina str., Stavropol, 355029, Russia
[2]Stavropol State Agrarian University, 12 Zootechnichesky, Stavropol, 355017, Russia

**Abstract.** The article discusses possible ways to use the Maple computer algebra system to obtain perfect conjunctive and disjunctive normal forms. Computer algebra systems make it possible to expand the possibilities for creating, applying and using mathematical models in the daily activities of engineers, researchers, and also contribute to improving the education of students of various training profiles. In addition, the use of computer technology can help to apply various methods and expand approaches to the representation of the studied objects (Boolean functions) of discrete form, including in education. Computer algebra systems such as Maple, Mathematica, etc. are the most adapted for studying various sections of discrete mathematics and mathematics in general. The possibilities of the Maple program are shown for use at the initial stages of mastering the discrete mathematics section of the university course - Boolean functions, in particular, for work on the construction and study of perfect conjunctive and disjunctive normal forms according to the truth table.

## 1 Introduction

In almost any higher educational institution, the course of discrete mathematics [1] or its section mathematical logic is a mandatory part of the curriculum and helps to better understand both the order of synthesis and analysis of the functioning of combinational devices, and the computer science course as a whole. However, one of the most persistent problems is that the subjects covered by the course of discrete mathematics and its mathematical logic section are not always directly related to the desire of students to study computer science more deeply. Many students are not always aware of the relationship between the topics covered in separate courses of mathematical and informational orientation. One of the goals of discrete mathematics, in our opinion, is to give students the opportunity to study more deeply the relationship between discrete mathematics and computer science.

Another problem observed with respect to the course of discrete mathematics or its mathematical logic section is the difficulty of understanding the individual issues being presented [2, 3]. Therefore, another goal of discrete mathematics is to improve the course by

* Corresponding author: olenevalexandr@gmail.com

providing interactive learning modules that increase interest and improve understanding of the material being presented.

Further, the possibility of increasing the visibility of the representation of objects of discrete mathematics is discussed, namely, the possibility of visual construction of perfect normal forms of Boolean functions. The article concludes with several reflections and views on improving the process of studying mathematics of discrete mathematics and its sections.

## 2 Method

It is well known that the Maple computer algebra system (CAS) is a mathematical system that operates and processes information in symbolic or algebraic form [4, 5]. Using Maple, it is possible to obtain an accurate analytical solution to mathematical problems [6], including integrals, systems of equations, differential equations, linear algebra problems [7], cryptography [8-10], discrete mathematics, including mathematical logic [11-13]. Maple also contains a large set of built-in functions, graphical procedures for evaluating and solving mathematical problems where exact solutions do not exist. In addition, the built-in programming language can be used to develop user functions and various types of applications. CAS Maple was first introduced in 1980, since then there have been many editions and versions. In this article, Maple 2020 is used, this is primarily due to the possibility of representing logical operations – mathematical symbols, which significantly expands the possibilities for studying and teaching mathematical logic. Expands the system's capabilities for constructing tests, assignments and exercises, as well as evaluating students' answers and speeches.

## 3 Numerical modeling

To demonstrate the operation of various simple devices, as well as to analyze logical functions, a visual representation in the form of a table is used. The table records the dependence of the output values of the function on the input values of the variables of the logical function. Such tables are called truth tables. However, obtaining the logical function itself is a rather laborious process and involves the possibility of obtaining a large number of errors. In our opinion, it is necessary to automate this process. Automation will allow you to perform these actions more clearly, quickly, and most importantly without errors, which allows you to proceed to the analysis and minimization of the logical function in a more reasoned manner in the future.

For example, consider the following truth table.

**Table 1.** Truth table.

| A | B | F(A,B) |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

The task is to obtain perfect conjunctive and disjunctive normal forms.

### 3.1 Obtaining a perfect conjunctive normal form (SCNF)

There are two rows in the table for which the function returns *false*. Let's imagine this table forming a set consisting of a list of values for A and B corresponding to these rows.

```
> restart;
>with(Logic); # connecting the library Logic
>Description_Tab_ist:= {[true, false], [true, true]};
```

This function will be the first argument of the procedure being created. To execute the created procedure, you will also need a list of variables used, and they will be the second argument of the procedure.

```
> Use_variables:= [A, B];
```

To form a logical expression representing this logical function, we will use the definition of a perfect conjunctive normal form - the conjunction of elementary disjunctions (maxterms). A maxterm is an elementary disjunction of maximum rank $r = n$, where $n$ is the number of variables used to describe a logical function. That is, it is necessary to describe each row in the table for which the function returns *false*, i.e. for each element in Description_Tab_ist, it is necessary to find the corresponding maxterm.

To create a maxterm associated with the Description_Tab_list element, use the following procedure:

```
> Maxterm:= proc (function::(list(truefalse)), variables::(list(symbol)))
local maxterm, i;
if nops(function) <> nops(variables) then error " Incorrect amount of data"
end if;
maxterm := NULL;
for i to nops(function) do if function [i] then
maxterm := Logic:-`&or`( maxterm, Logic:-`&not`( variables [i]))
else maxterm := Logic:-`&or`( maxterm, variables [i]) end if end do;
return maxterm
end proc;
```

We will check the correctness of the developed function:

```
> Normalize(Maxterm ([true, true], [x, y]));
```

Received response:

$$(\neg x) \vee (\neg y)$$

The result is correct.

To reproduce a perfect conjunctive normal form for a given Boolean function, it is necessary to form a conjunction of maxterms for a concrete table (in our case, this is the Description_Tab_ist procedure). The procedure below takes a tabular representation and a list of variables as arguments. First, it checks whether an empty set has been passed, and, if so, returns the message "Incorrect amount of data", which corresponds to the presence of an error in the specified data.

Procedure for obtaining a perfect normal conjunctive form:

```
> SKNF := proc (Table::(set(list(truefalse))),    Var::(list(symbol)))
local Pr_max, value, maxis;
if Table = {} then return false end if;
Pr_max:= NULL;
for Var in Table do
maxis := Maxterm (value, Var);
Pr_max:= `&and`( Pr_max, maxis) end do;
return Pr_max
end proc;
```

The result obtained for the truth table is 1.

```
> K := Normalize(SKNF (Description_Tab_ist, Use_variables), form = CNF);
```

$$K := (A \vee B) \wedge (A \vee (\neg B))$$

We will check the correctness of the correspondence of the obtained result, that is, the correspondence of the received form to the truth table. To do this, use the Logic – TruthTable library function:

> TruthTable(%, output = Matrix)

| | $A$ | $B$ | $value$ |
|---|---|---|---|
| 1 | *false* | *false* | *false* |
| 2 | *false* | *true* | *false* |
| 3 | *true* | *false* | *true* |
| 4 | *true* | *true* | *true* |

**Fig. 1.** The truth table for the SKNF of the logical function presented in Table 1.

The analysis of the received truth table confirms the correctness of the received SKNF.

### 3.2 Obtaining a perfect disjunctive normal form (SDNF)

To form a logical expression representing this logical function, we will use the definition of a perfect disjunctive normal form - disjunction of elementary conjunctions (minterms). A minterm is an elementary conjunction of maximum rank $r = n$, where $n$ is the number of variables used to describe a logical function. That is, it is necessary to describe each row in the table for which the function returns *true*, i.e. for each element in Description_Tab_ist, it is necessary to find the corresponding minterm.

To create a minterm associated with the Description_Tab_list element, use the following procedure:

```
> Mintherm := proc (function::(list(truefalse)), variables::(list(symbol)))
local mintherm, i;
if nops(function) <> nops(variables) then
error " Incorrect amount of data " end if;
mintherm := NULL;
for i to nops(function) do if function [i] then
mintherm := Logic:-`&and`( mintherm, variables[i]) else
mintherm := Logic:-Normalize(Logic:-`&and`( mintherm, Logic:-`&not`( variables
[i]))) end if   end do;
return mintherm
end proc:
```

We will check the correctness of the developed function:

> Normalize(Mintherm ([true, true], [x, y]));

Received response:

$$x \wedge y$$

The result is correct.

To reproduce a perfect conjunctive normal form for a function, it is necessary to form a disjunction of minterms for a specific table (in our case, this is the Description_Tab_ist procedure). The procedure below takes a tabular representation and a list of variables as arguments. First, it checks whether an empty set was passed, and if so, returns an expression with an incorrect number of variables, i.e. it shows the presence of an error.

Procedure for obtaining a perfect normal disjunctive form:

```
> SDNF := proc (Table::(set(list(truefalse))), Var::(list(symbol)))
local Pr_min, value, mint;
```

```
if Table = {} then return false end if;
Pr_min:= NULL;
for value in Table do mint := Mintherm (value, Var);
Pr_min:= Normalize(`&or`( Pr_min, mint)) end do;
return Pr_min
end proc:
```

The result obtained for the truth table 1.

```
> SDNF (Description_Tab_ist, Use_variables);
```

$$(A \wedge B) \vee (A \wedge (\neg B))$$

We will check the correctness of the correspondence of the obtained result, that is, the correspondence of the received form to the truth table. To do this, use the function of the Logic - TruthTable library:

```
> TruthTable(%, output = Matrix)
```

| | $A$ | $B$ | $value$ |
|---|---|---|---|
| 1 | $false$ | $false$ | $false$ |
| 2 | $false$ | $true$ | $false$ |
| 3 | $true$ | $false$ | $true$ |
| 4 | $true$ | $true$ | $true$ |

**Fig. 2.** The truth table for the SDNF of the logical function presented in Table 1.

The analysis of the received truth table confirms the correctness of the received SDNF.

Additionally, you can use the Logic library function to verify the correctness of the received forms:

$$Equivalent(\mathrm{K}, (A \wedge B) \vee (A \wedge (\neg B)))$$

Using the above procedures, the problems of obtaining perfect conjunctive or disjunctive normal forms of a Boolean function are solved according to the available (given) truth table. The results obtained are used in the future to minimize the Boolean function and analyze the results obtained.

## 4 Conclusions

Computer algebra (CAS) systems such as Maple, Matlab, Mathcad and Mathematica can be used as powerful assistants for performing symbol manipulation and calculations in discrete mathematics or its section - mathematical logic. It is suggested that these systems will be useful to students and postgraduates in the field of mathematics, engineering and physics, and will help to keep track of details during complex calculations. This work only shows how the symbolic mathematical system Maple can be used to help in understanding both theoretical and computational aspects of some topics of mathematical logic.

Truth tables and the construction of perfect forms of logical functions are familiar topics for second- and third-year students and are considered the foundation for understanding the construction of both simple and complex devices and systems, as well as analyzing the operation of various digital devices. However, these topics are considered difficult for students to understand. Using the help of CAS Maple to describe only a part of the truth table, as well as preparing data for further analysis of the truth table gives students more time to study other important properties and principles. Thus, the use of Maple SKA should arouse enthusiasm among both students and teachers.

Preliminary results of this study show an improvement in the ability of students to understand the order of obtaining perfect forms according to the truth table.

Preliminary results of this study show an improvement in the ability of students to understand the order of obtaining perfect forms according to the truth table.

We thank to the members of the editorial board for their patience, timely and complete technical editing, language editing and editing of the text of the article.
We welcome any questions, comments and suggestions for improving the article.

## References

1. K.H. Rosen, *Discrete Mathematics and Its Applications* (New York: Published by McGrawHill, 2012)
2. S. Abramovich, J. Burns, S. Campbell, A.Z. Grinshpan, IMVI Open Mathematical Education Notes **6**, 65-106 (2016)
3. P.W. Tompson, M. Artigue, G. Törner, E. de Shalit, *Collaboration between mathematics and mathematics education Mathematics and Mathematics Education*: *Searching for the Common Ground* (Springer, Dordrecht, 2014)
4. M.B. Monagan, K.O. Geddes et al., *Maple Introductory* (Programming Guide Maplesoft a division of Waterloo Maple Inc, 2009)
5. R.J. Lopez, *Maple via Calculus: A Tutorial Approach* (Springer Science & Business Media, 2012)
6. H. Hamid, N. Angkotasan, A. Jalal, D. Muhtadi, Sukirwan, J. Phys.: Conf. Ser. **1613**, 0120252020 (2009)
7. Y.L. Ningsih, R. Paradesa, J. *Phys.:* Conf. Ser. **948**, 012034 (2018)
8. J.L. Pardo Gómez, *Introduction to Cryptography with Maple* (New York: Springer, 2013)
9. R.E. Klima, N.P. Sigmon, *Cryptology: Classical and Modern with Maplets* (New York: Chapman & Hall/CRC, 2012)
10. K.A. Kirichek, E.V. Potekhina, N.V. Grivennaya, O.V. Pelikh, AIP Conf. Proc. **2647**, 050004 (2022)
11. A.A. Olenev, K.A. Kirichek, E.V. Potekhina, O.V. Pelikh, J. Phys.: Conf. Ser. **1691**, 012097 (2020)
12. K.T. Tyncherov, M.V. Selivanova, J. Phys.: Conf. Ser. **1661**, 012086 (2020)
13. E.M. Petlina, A.V. Shuvaev, N.V. Grivennaya, A.N. Khabarov, *Maple Information Tools in the Study of Mathematical Logic Questions*, in Hybrid Methods of Modeling and Optimization in Complex Systems, vol 1. European Proceedings of Computers and Technology. European Publisher (2023)